(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 811 942 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
10.12.1997 Bulletin 1997/50

(51) Int. Cl.[6]: **G06F 17/60**

(21) Application number: **97109222.6**

(22) Date of filing: **06.06.1997**

(84) Designated Contracting States:
AT BE CH DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE

(30) Priority: **07.06.1996 US 660488**

(71) Applicant:
Cyber Media, Incorporated
Santa Monica, CA 90405 (US)

(72) Inventors:
- **Cheng, William**
 San Gabriel, CA. 91775 (US)
- **Hwang, Kenneth**
 Los Angeles, CA. 90049 (US)
- **Kannan, Ravi**
 Los Angeles, CA. 90024 (US)
- **Katchapalayam, Babu**
 Los Angeles, CA. 90049 (US)
- **Liu, Bing**
 Alhambra, CA. 91801 (US)
- **Narasimhan, Balaji**
 Culver City, CA. 90230 (US)
- **Ramanujam, Gopal**
 Los Angeles, CA. 90025 (US)
- **Tran, Jonathan**
 Alhambra, CA. 91803 (US)

(74) Representative:
Liesegang, Roland, Dr.-Ing. et al
FORRESTER & BOEHMERT
Franz-Joseph-Strasse 38
80801 München (DE)

(54) Automatic updating of diverse software products on multiple client computer system

(57) A system and method update client computers of various end users with software updates for software products installed on the client computers, the software products manufacturered by diverse, unrelated software vendors. The system includes a service provider computer system, a number of client computers and software vendor computer systems communicating on a common network. The service provider computer system stores in an update database information about the software updates of the diverse software vendors, identifying the software products for which software updates are available, their location on the network at the various software vendor computer systems, information for identifying in the client computers the software products stored thereon, and information for determining for such products, which have software updates available. Users of the client computers connect to the service provider computer and obtain a current version of portions of the database. The client computer determines that software products stored thereon, and using this information, determines from the database, which products have updates available, based on product name and release information for the installed products. The user selects updates for installation. The selected updates are downloaded from the software vendor computer systems and installed on the client computer. Payment for the software update and the service is mediated by the service provider computer. Authentication of the user ensures only registered users obtain software updates. Authentication of the software updates ensures that the software updates are virus free and uncorrupted. Changes to the client computer during installation are monitored and archived, allowing the updates to be subsequently removed by the user.

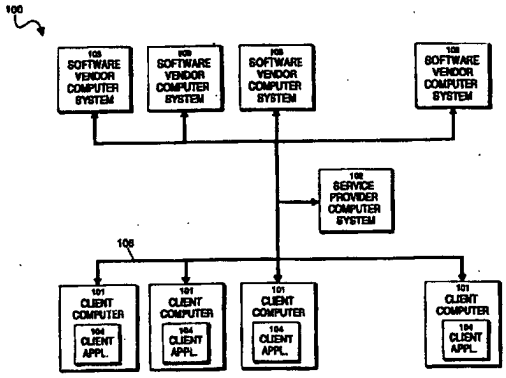


FIGURE 1

EP 0 811 942 A2

Description

BACKGROUND

Field of Invention

The present invention relates to systems and methods for computer-based customer support, and more particularly, to systems, methods, and products for automatically updating software products from diverse software vendors on a plurality of end-user, client computer systems.

Background of Invention

The typical personal computer contains various categories of software products, such as operating system files, utilities, applications, and device drivers, code libraries, and other forms of computer readable or executable information. In some of these categories, such as applications, the personal computer may contain numerous programs in various subcategories. For example, a user may have one or two word processing applications, several graphics applications, and numerous games. Most of these products will come from different software vendors. As used herein "software vendors" includes any entity that distributes software products, even if the entity also manufactures or distributes hardware or other non-software products. These software vendors frequently improve their products, by adding new features, or by fixing known problems, and make these software updates available to their users. These updates may or may not be free.

There are at least three significant problems that the vendors and users face in attempting to provide these updates to the user. First, vendors face difficulty and costs in attempting to inform users of their products that the updates are available, and users experience similar difficulties in attempting to ascertain what updates are available. Vendors typically send out mailings to registered users, place advertisements in relevant trade journals and magazines, and engage in other promotional activities.

For all of these efforts, many users may remain unaware of the many software updates applicable to their systems until they encounter problems and contact the vendors' technical support organizations. Other users only learn about updates by searching the Internet or on-line services for solutions to their technical problems. Just the shear magnitude of the problem of updating all software products can be overwhelming. Given that a user will have many software products from numerous vendors on her computer, it would be nearly impossible for the user to frequently monitor all of the available distribution channels, journals, Internet forums, and the like, to determine for which of the many software products there are updates available.

For example, some vendors maintain sites on the World Wide Web, or electronic bulletin boards (BBS's) that include information about current updates and products, and enable a user to download such updates. However, such sites are obviously dedicated to a single software vendor, and provide information only about that software vendor's products, and certainly not about the products of numerous other vendors that may be interest to a given user. Thus, the user would have to search the Internet, and possibly online services, to determine which vendors have such sites. The user would likely to have visit each of these sites individually and determine what software updates are available from each of them. Similarly, even though some on-line services include forums or other mechanisms where users can learn about available updates, this still places the burden on the user to actively seek out this information. Directories or search engines on the Internet, such as Excite, Yahoo, Lycos, or Infoseek merely provide links to software vendor sites, but do not generally attempt to systematically determine which software updates are available, and provide this information to the user, let alone actually update the software on the user's machine.

Another problem is that even once an update has been identified, there is the need to install it in the user's computer. Many users purchase the software updates by mail order, or the like, and receive them on floppy diskettes. Other users may download the software updates via Internet from the computers of the software vendors, or from on-line services. In any of these cases installing a single update can be a tedious, time-consuming and error-prone process for many users due to the various formats and installation procedures required. Installing updates for all of the numerous software products on a user's system on a regular basis would be even more difficult and time consuming for the typical user.

Finally, many users have concerns about their privacy, and are often resistant to revealing complete information about their software configurations to one or more vendors. However, even for a single vendor, information about which of the vendor's products are installed on a user's computer system, and system configuration information is necessary for determining which updates are applicable to the user's computer system. For example, a certain software update to an accounting program from vendor A might be applicable if the user has a printer from vendor B, and a different software update is applicable if the printer comes from vendor C. The user might not want to let each vendor know about all the components on their system, but this configuration information is necessary to ensure the correct software updated is installed. Still, users are resistant to the prospect of a single vendor storing information profiling the software

components that reside on their computer systems.

In summary, from the perspective of an individual vendor, the problems are identifying and notifying every user of the vendor's software of the availability of updates to the software on a timely and useful basis, and ensuring that the proper software updates are installed. From the perspective of the individual user, the problems are systematically and easily identifying which updates are currently available for every piece of software on her system, and resolving the technical difficulties in obtaining and installing such updates.

Accordingly, it is desirable to provide a system that automatically determines which software updates from numerous diverse software vendors are currently available, and which are applicable to a given user's computer system, and installs such user selected ones of such updates on the user's computer. Further, it is desirable to provide such a system without abridging the privacy of users by obtaining and storing system profile information.

SUMMARY OF THE INVENTION

In accordance with one aspect of the present invention, there is provided a system and method that automatically updates software components from numerous diverse software vendors on the computer systems of a plurality of end users. The system includes at least one database that stores software update information for a plurality of software products manufactured by diverse software vendors. The database is maintained by a service provider on a service provider computer system. Alternatively, the database may be maintained by a set of software vendors of the software products in association with the service provider. The software update information in the database specifies the software update program or files and their network location on the computer system of the software vendors, which computer systems are connected over the network to the service provider computer system. The database further stores information that describes an installation process for installing the software update on a user's computer.

In the preferred embodiment the network is the Internet to which the diverse software vendor computer systems are connected, and make available updates for their software. The network location provided in the software update information is specified by one or more universal resource locators (URL). Thus, the update database does not store the software updates themselves, but information that is used to access the software from computer systems of the original vendors. In addition, the software update information includes a format description associated with specific installation programs or actions needed to install the software update.

On each user computer, or synonymously client computer, operating in accordance with one embodiment of the invention there is provided a client application that periodically connects over the network to the update database of the service provider computer system. The client application automatically downloads a portion of the database to the client computer, preferably to update a mirror of portions of database. From client database, the client application determines which software updates are applicable or relevant to the user's computer. This is preferably done by first determining the products that are installed in the client computer, and determining for each of these whether there is an update available for a more recent version of the software product than that installed on the client computer. The applicable software updates are identified to the user.

The user selects various software updates for installing on the client computer. Either the client application or the service provider computer then uses the network location specified in the database to connect to the computer system of the software vendor and download the software update from there to the client computer. The client application uses the format information to determine the appropriate installation process associated with the software update, and installs the software update using the proper installation process. During the installation process, the client application monitors all changes made to the client computer, such as the deletion, addition, or alteration of files or directories. The client application stores the state of the client computer prior to these changes. This may be done by saving copies of files to be altered or deleted, and noting pathnames of files or directories to be added. Once the installation is complete the stored state information, such as the copied files, is compressed and archived, and identified as being associated with the just installed software update. This stored state information allows the client application to "undo" the installation and restore the client computer to its state prior to the installation, including restoring any files that were deleted or altered. Multiple "undoes" of multiple installations may be accomplished.

Where the vendor charges for the software update, the service provider computer may intermediate between the software vendor's computer and the client computer to enable an electronic payment by the user to the software vendor. When the electronic payment is authorized, the client application downloads the software update and completes the installation as described. Additionally, the service provider computer may authenticate the software update and check to ensure that it is not corrupted, for example by computer viruses.

The service provider periodically updates the update database, monitoring URL information to ensure that it correctly identifies the location of software updates, identifies new software updates, or new products for inclusion in the database, and the like.

The above system allows numerous users to periodically and automatically update the software products on their computers from diverse software vendors through a single, update mechanism. The users need not invest the time and energy to identify currently available updates, nor engage in the potentially difficult process of manually (even electron-

ically) obtaining and installing the software updates, and properly configuring their computer systems. Rather, all of the relevant information about the currently available updates is maintained for subscribing users in the service provider's database. Further, the above system provides these benefits without directly storing the software updates themselves, which would be undesirable for the service provider due to vast memory requirements needed for handling software updates from hundreds, or potentially thousands of software vendors, and the difficulty of ensuring that all such software updates were current.

As a further aspect of the invention, the update database stores promotional information for various software products in addition to the software update information. The promotional information includes product literature, advertising literature, technical information, product demonstration software and the like. This information is categorized into a number of product categories. Like the software update information, the promotional information itself is not stored in the update database, but rather, a network location is stored for each item of promotional information.

The client application determines on the client computer the product categories of the software products therein, and then selects promotional information from certain product categories, and retrieves this promotional information from the network location information associated with it. This enables the client application to direct promotional information to the client computer based on the software products thereon, and hence the user's likely product interests, without 1) storing large quantities of promotional information directly; and 2) disclosing the actual contents of the user computer to third parties.

The update database of software update information is preferably maintained by a supplier of the client application, who also maintains the service of providing software updates. This database is created by engaging with numerous software vendors to provide information for software updates of their products to the service provider. Preferrably, the software vendors provide the service provider information describing the software update, including version information, file formats, configuration information, and network location.

Alternatively, software updates may be identified by systematically and periodically searching the Internet to identify the software vendors who are providing updates to their software products. These updates are then downloaded from the software vendor's Internet sites, and one or more network locations (URL) are identified for obtaining the download. The downloaded software update is then installed on one or more target computers as typically configured by end users. The format of the software update is determined, along with the network location of the software update, and other descriptive information, such specific configuration actions to be taken during installation of the software update, useful descriptive text, and the like.

In another aspect of the present invention there is provided a client application that interacts with the update database. The client application is provided on each of a plurality of client computers. The client application is executed on a periodic basis and connects to a update database as described above to perform the operations of downloading the portions of the database, determining the relevant software updates to be retrieved, retrieving same from their network locations, installing the software updates on the client computer, and removing installed updates if desired.

In yet another aspect of the present invention, information about software updates or software products may be provided by the service provider to a client computer by email, upon request for email notification by a user. Users who subscribe to the service provider's service indicate the software products in which they are interested, either directly, or indirectly by their actions in using the service. As new software updates, or software products become available, the service providers determines which one match a particular user's interests, and notifies the user of such updates or products by email. The notification includes the network location of where the software product or software update is available. The user can then approve the installation of the product or update, and use the client application to download and install the software update or product.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is an illustration of a system for providing software updates in accordance with the present invention.

Figure 2 is a flowchart of the overall method for providing software updates to a client computer in accordance with the present invention.

Figure 3 is an illustration of a user interface for registering a new user of the updating service.

Figure 4 is an illustration of a user interface for selecting software updates for installation.

Figure 5 is an illustration of a user interface for confirming installation of a software update.

Figure 6 is an illustration of a user interface for undoing an installation of a software update.

Figure 7 is an illustration of software architecture of the service provider computer system.

Figure 8 is one embodiment of a schema for the update database of the service provider computer.

Figure 9 is an illustration of the software architecture of an client computer.

Figure 10 is a flowchart of further details of analyzing the client computer, determining software updates, and displaying update information.

Figure 11 is a flowchart of the operation of the install monitor.

Figure 12 is a flowchart of the operation of the URL monitor.

Figure 13 is an illustration of a user interface for registering a software update into the update database.
Figure 14 is one embodiment of a schema for the user profile database.
Figure 15 is one embodiment of a schema for the advertising information database.
Figure 16 is a flowchart of the operation of the recovery module.
Figure 17 is an illustration of a user interface for registration a sofware product into the update database.

DETAILED DESCRIPTION OF THE INVENTION

System Architecture

Referring now to Figure 1, there is shown the architecture of one embodiment of a system for updating diverse software products on user's computers in accordance with the present invention. In system 100, there are a plurality of client computers 101 communicatively coupled by a network 106 to a service provider computer 102. A number of software vendor computers 103 are also communicatively coupled over the network 106 to the service provider computer 102. The network 106 is preferably the Internet, or other similar wide area network.

Each client computer 101 is operated by an end user, and typically has a number of software products installed thereon, such as applications, drivers, utilities and the like. In accordance with the present invention, the client computers 101 includes a client application 104 that communicates with the service provider computer 102 to obtain software updates of software products installed on the client computer 101. The software architecture of a client computer 101 and client application 104 is further described below with respect to Figure 7.

Each software vendor computer 103 coupled to the service provider computer 102 stores software update information, software products, information files, and the like. The software update information includes applications, binary files, text files, and the like, for updating software products installed on client computers 101, and advertising or other information about such products useful to users for evaluating potential software for updating. Other types of information useful to providing product support, technical service, or the like may also be beneficially provided. In addition, the software vendor computers 103 provide mechanisms for controlling distribution and payment of software updates, such as credit card payment front ends, code authentication and verification subsystems, and the like. These various mechanisms are understood in the art. For example, payment mechanisms may be implemented in compliance with various credit card or debit systems, as known in the art. Likewise, authentication and verification may be implemented using conventional encryption techniques.

In a preferred embodiment, the network 106 is the Internet, and more specifically, the World Wide Web portion thereof. The various computers thereby support the protocols for FTP, and HTTP, and provide for the display and rendering of HTML, VRML, or other text or interface description languages. Each computer 101, 102, 103 has a IP address that specifies its location on the network 106, thereby allowing such computers to communicate with each other in a conventional manner. Files, such as executables, binaries, and text files are identified within the various computers by universal resource locators (URLs) as known in the art.

Overall System Operation

Referring now to Figure 2, there is shown an overall flow diagram of the process of updating a single client computer 101 in accordance with the present invention. The process here is described with respect to a single client computer 101. Given the client-server nature of the system, those of skill in the art understand that numerous other individual client computers 101 may interact with the service provider computer 102 in parallel.

The update process 200 is typically initiated on the client computer 101. The user may manually initiate the process, or it may occur automatically, for example at preset periods, such as once a month. Alternatively, the process may be initiated by the service provider computer 102 prompting the client computer 101 at various intervals, or in response to particular events.

In each case, the user logs in 201 to the service provider computer 102 with the client application 104 in a conventional manner, providing a user ID, a password, and the like. This information may be manually entered by the user via the client application 104, or more preferably, stored within the client application 104, and automatically provided once a connection between the client computer 101 and service provider computer 102 is established. If the user is not registered, then the service provider computer 102 in conjunction with inputs by the user, registers 202 the new user of the system. Figure 3 illustrates a basic user interface 300 for registering the user. The user identifies himself or herself by name 301 and selects a password 303. The user may also provide a mailing address 305 and a payment mechanism such as a credit card data 311, including a credit card number and expiration date, to pay for the services and for any for-fee software updates that the user may access in the course of using the service provided by the service provider computer 102. An email address 307 is entered to allow the service provider to contact the user by email. The user may select check box 309 to indicate that they want to be notified by email when new software updates are available for software products installed on their computer. When the registration process 202 is completed, the service provider com-

puter 102 returns a unique registration number to the user. This number may be stored on the client computer 101 and used during subsequent logins to identify the user to the service provider computer 102.

The registered users are authenticated 203 by the service provider computer 102, using conventional authentication mechanisms, such one or more passwords, digital signature, certificates, or the like. Authentication ensures that only users who are properly authorized by the service provider can obtain updates for software products.

The client application 104 then analyzes 204 the client computer 101 to determine a list of installed software products. The list of installed software products typically includes applications, system utilities, drivers, and other executables or resources. These software products will typically be from numerous diverse software vendors, a number of whom will maintain software vendor computers 103 on the network 106.

For each of the installed software products on the list, the client application 104 determines 205 if there is an applicable, or relevant update for the software product. This determination is made in consultation with the service provider computer 102, which maintains, as further described below, a database including a list of available software updates for numerous software products of diverse software vendors.

The client application 104 displays 206 the list of applicable software updates to the user, for review and selection thereof of updates for purchase and installation. Figure 4 illustrates a sample user interface display 400 of applicable software updates. This display 400 includes the name 401 of each software product identified on the client computer 101, and remarks 403 displayed next to the name indicating whether the software product is already up-to-date, that is, there are no applicable updates, or, if the product is not current, the list of applicable updates (which may be for the software product itself, or for related products). In those cases where there is an applicable update, the remarks 403 briefly indicate the nature of the software update. In the example of Figure 4, the remarks 403 for the software product Quicken 5.0® by Intuit Inc., indicates a update to provide new features. The user may obtain additional information by selecting a name or remark of a particular software product. The selected product name and remark is highlighted, as shown in Figure 4, and the information about the software update is displayed 207 in an information window 405. This information may be stored in the service provider computer 101, or obtained directly from the software vendor computers 103 as needed using URLs associated with such information. The user may limit the list to only those software products that need updating, rather than all installed software products, by selecting check box 407.

The user may select one or more software products to update. To update one of the software products, the user selects the software product for update by selecting (e.g. double-clicking) the line including the software product, or by single clicking on the line, and then clicking the retrieve button 409. The user may select more than one software update by holding the control key on the keyboard down while single-clicking on the name of each desired software update, followed by selecting the retrieve button 409. When all the desired updates have been selected, the user may click on the continue button 411 to begin the installation process.

For each selected software update, the client application 104 performs an installation process 208. Referring to Figure 5, the client application 104 displays information 505 for a selected software update, and provides the user the opportunity to confirm 501 or cancel 503 the installation. If confirmed, the client application 104 downloads 209 the software update, along with installation information, such as installation programs, files, and the like. This downloading may be directly from the software vendor computer 103, using the URL data stored in the service provider computer 102 for the location of the software update on the network 106.

In conjunction with the downloading process 209, a payment transaction 210 may be conducted whereby the user of the client computer 101 pays for the software update if it is not a free update. The service provider computer 102 may intermediate in this transaction, or merely initiate the transaction by connecting the client application 104 to the computer 103 of the software vendor of the update. If payment information, such as credit card numbers, are stored in the client application 104, then this information maybe provided by the client application 104 to the software vendor computer 103.

Once the download and applicable payment are complete, the software update is physically installed on the client computer 101. Each sofware update is associated with information that describes the particulars for the installation, such as configuration, decompression or other information. The installation is performed in conformance with such information.

In the preferred embodiment, the client application 104 executes 211 an install monitor prior to actually installing the software update. The install monitor, as further described below, records the changes made to the client computer 101 as a result of the installation of the software update. This information is archived by the install monitor and allows the user to "undo" or remove any number of installations, and restore the client computer 101 to its state prior to each such installation. Accordingly, the client application 104 performs 212 the installation, executing any necessary decompression, installation, or setup applications necessary to install the software update. During the installation process 212 the install monitor records 213 any changes made to the system configuration, including changes to various configuration files, additions or deletions of files, and additions or deletions of directories. The changes may be recorded in a variety of manners, such as building descriptions of the modifications of the files, or alternatively, storing copies of files prior to their alteration or deletion. Once the installation is complete, the install monitor archives 214 the changes. This process 208 is repeated for each software update to be installed.

Once all of the software updates have been installed, the client applications 104 logs out 215 of the service provider computer 102, and any necessary payment information for the user may be updated, such as payment based on the number of software updates purchased, the online connection time, and the like. Alternatively, no payment may need to be directly made, as the cost of the service may be included in the cost of the software update charged by the software vendor, who then pays the service provider for the service of coordinating and linking end users to the software vendor's computer system 103.

At some subsequent point, the user may decide to undo a previous installation, for example, due to dissatisfaction with the software product. The user may use a recovery feature of the client application 104 to undo 216 the installation. A sample user interface 600 for the recovery function is illustrated in Figure 6. The user interface 600 includes a field 601 indicating the previous update to be removed as selected by the user, along with an information window 603 describing the software update. The user confirms the removal of the software update by selecting the undo button 605, or may cancel with cancel button 607. The recovery function deletes the files installed for the software update, and using the archived information created by the install monitor during the installation of the product, restores the client computer system 101 to its configuration immediately before the installation of the product. This process 216 includes deleting files and directories that were added, restoring files and directories that were deleted, and restoring files that were otherwise changed. In one preferred embodiment, the recovery function is able to undo any installation in a given series of installations, accounting for changes to the configuration of the client computer 101 after a particular installation. In another preferred embodiment, the recovery function undoes installations in the reverse order of their installation. If any payments were originally required from the user for the cost of the software update and the associated service of downloading and installing it, the payments may be credited back to the user when the user undoes the installation.

Service Provider Computer

Referring now to Figure 7, there is shown one embodiment of the service provider computer 102 in accordance with the present invention. In terms of hardware architecture, the service provider computer 102 is conventional server type computer, preferably supporting a relatively large number of multiple clients simultaneously for requests for data and other processing operations. The service provider computer 102 includes one or more conventional processors in a processor core 723, and a suitable amount of addressable memory 700, preferably on the order of 18-64 Mb. The service provider computer 102 may be implemented with an Intel-based computer including one or more Pentium® processors, or other more powerful computer, such as various models of Sun Microsystems' SparcStations using UltraSparc® processors. The service provider computer 102 executes a conventional operating system 721, such as Windows NT® from Microsoft Corp., or one of various UNIX-based operating systems, such as Sun Microsystems' Solaris 2.5. The service provider computer 102 further includes a network communication protocol layer 719 that implements the necessary TCP-IP communication functions for connecting to the network 106 and communicating with other computers.

In accordance with the present invention, the service provider computer 102 includes a number of executable components and database structures useful for managing the software update interactions with the client computer 101 and the software vendor computers 103. These components include a security module 701, a communications module 703, a payment module 705, database modification tools 707, a update database 709, a user profile database 711, a reporting tools module 713, a URL monitor module 715, an advertising/information database 717, and an activity log 718. The update database 709 is described here; the remaining components are described further below.

Update Database

The update database 709 maintains information identifying a large number of software products, information about the software updates that are available from the diverse software product vendors for these software products, information for identifying software products installed on a client computer 101, and for uniquely distinguishing the versions and names of installed software products.

In one embodiment, the update database 709 does not itself store the software updates, but rather stores information, such as URLs, that allows the service provider computer 102 or the client computers 101 to directly access the software updates from the software vendor computers 103. This implementation is chosen for several reasons. The system 100 is designed to provide software updates for large numbers of software products, on the order of hundreds, and perhaps thousands of products. In this situation, extremely large amounts of storage would be required to store the relevant files. Further, by not storing the software updates themselves, but only links to the software vendor computers 103, the service provider does not have to make sure that the software updates themselves are always current, but need only maintain the link information, which is administratively easier. In another embodiment, the software updates are stored in the updated database 709. This implemenation is useful, for example, to facilitate synchronization of updates of the database 709 itself with the releases of new software updates for software products, thereby ensuring that the entries in the database 709 are consistent with the current releases of new software updates.

Finally, the update database 709 may also store information describing an installation process for installing a software update. This information may include particular configuration, file format, or other data useful to performing the installation of the software update the client computer 101. This information, if present, may be provided to the client computer 101 to use during the installation of the software update.

The update database 709 may be implemented in a variety of ways. Referring now to Figure 8 there is shown one implementation of the update database 709, illustrated as a schema for a relational database. In this embodiment, the update database 709 includes 4 tables: a method table 801, a product locator table 803, an product table 805, and an update table 807. Figure 9 illustrates a flowchart of the process of analyzing the client computer 101 using the tables of the update database 709.

The method table 801 maintains information identifying various methods of analyzing a client computer 101 to determine which software products are installed thereon. The method table 801 includes scan methods 811 and parameters 812. The various scan methods 812 are designed to cover the variety of different facilities of a client computer 101 that may identify the installed products. For example, in a client computer 101 using Microsoft's Windows95 or Windows NT operating system, there is provided a Registry which is designed to maintain indicia of installed software products. The Registry includes various methods that can be called to return information about the software products identified therein. Some of these methods are listed in the scan methods 811. The parameters 812 are arguments to the Registry methods, for example, identifying specific aspects of the Registry to be searched.

While compliance with the Windows95 standard requires that a software vendor's installation procedure should update the Registry, not all software vendors comply. In this case, information identifying the installed software products is also maintained in the config.sys, system.ini, and the autoexec.bat files. Also, a client computer 101 may be using Microsoft Corp.'s MS-DOS or Windows 3.1 operating systems, which do not use the Registry. Accordingly, the scan methods 811 include methods for reviewing these system files and returning indicia of the installed software products.

Each of the scan methods 812 return indicia of the installed products in the form of a number of strings, here scan_string. Each scan_string identifies a product name or file name, or some other data. However, a scan_string may not uniquely identify a product. For this reason, the scan_string is resolved by the product locator table 803.

The product locator table 803 associates individual scan-strings 813 with a product name 815, instructions 816 for determining a version number or release number, and one or more constraints 814. The constraint is a rule that uniquely identifies the product given contextual information for the product where there are two entries having identical scan_strings. Constraints include specific directories that include the product, additional entries in the system configuration file, the Registry or the like. If the specified information in these various locations matches the constraint values, then the product name associated with the constraint is the correct product name for the scan_string. In one embodiment, the constraint 814 is an executable procedure that retrieves information in these various locations, and determines from this information whether the product name is a match with the scan_string, according to whether the specified details of the constraint are found in the client computer 101.

Since some of the installed software products will be in their most current version, it is not necessary to update all software products installed on the client computer 101. Rather, from the list of installed software products, further analysis (205, Figure 2) determines for which of these software products is there an applicable software update. A software update is applicable to a client computer 101 if version of the software update is more recent than the version of the installed software product.

Since not all of the software products installed on a client computer 101 need to be updated, the determination of the applicable software updates is usefully made with the product table 805. The product table 805 associates a product name 815 and a particular release 818 with an update ID 819 identifying a software update for that version of the product. The new version number 820 specifies the new version that would be produced by applying the software update specified by the update ID 819 to the software product identified by the product name and release number. The latest field 821 specifies (Y/N) whether applying the software update would bring the product to its most up-to-date version.

Finally, the update table 807 stores the information necessary for performing the software update itself. This table is usefully keyed by the update ID 819. For each update, there is provided a URL list 823 which contains URLs for the various sites that store the actual binary files for the software update, typically the software vendor computer system 103, and potentially mirror sites. The URL list 823 is comprised of a number of URL entries, each URL entry having a URL and a timestamp of the last time the URL was validated, and flag indicating whether the URL is valid. This allows the URL monitor 715 to ensure that current URL information is maintained in the database.

The current cost 824 of the software update is also stored to provide the user with cost information for the software update.

The format 825 specifies the file format of the software update files, and thereby indicates the type of processing needed to install the software update files. In one embodiment, there are six formats and accompanying installation procedures:

Table 1

Format	Installation Procedure
zip	1) Unzip file with unzip.exe 2) Run install.exe
zip	1) Unzip file with unzip.exe 2) Run setup.exe
self-extracting archive	1) Execute file to extract 2) Run install.exe
self-extracting archive	1) Execute file to extract 2) Run setup.exe
file.exe	1) Execute file for self extraction and installation.
unknown	1) use script information to perform installation.

With respect to unknown or custom formats, the update table 807 stores in the script 826 either a handle to a custom installation program that is provided either by the software vendor for the update, or by the service provider. In addition, the script 826 also stores information about any conditions that are required for the installation, such as turning off anti-virus programs, or other conflicting programs during the installation process.

The description 827 field stores data associated with a description of the software update, such as describing the product features. The description is preferably a URL to a file on the software vendor computer system 103 that contains the description information. Again, the actual text need not be stored here, but merely a link to where that information is available on the network 106.

The update database 709 has been described as a set of tables. Alternatively, the update database 709 may be implemented in an object oriented framework with each table being a class, and the fields of the tables being attributes and methods of the class. The class type is then usefully defined by the primary key of the table.

Client Computer

Referring now to Figure 9, there is shown an illustration of the hardware and software architecture of a client computer 101. A client computer 101 is of conventional design, and includes a processor core 918, an addressable memory 900, and other conventional features (not illustrated) such a display, a local hard disk, input/output ports, and a network interface. The display is of conventional design, preferably color bitmapped, and provides output for a user interface for various applications, such as illustrated in Figures 3-6. The input/output ports support input devices, such as a keyboard, mouse, and the like, for inputting commands and data. The network interface and a network communication protocol 916 provide access to remotely situated mass storage devices, along with access to the Internet, with a TCP-IP type connection, or to other network embodiments, such as a WAN, LAN, MAN or the like.

In the preferred embodiment the client computer 101 may be implemented on a Intel-based computer operating under Microsoft Windows 3.1 or Windows95 operating system 917, or equivalent devices. The client computer 101 includes some number of configuration files 915, such as the Windows95 Registry the system.ini, config.sys and other files.

The client computer 101 further has installed thereon software products in the form of applications 912, operating system utilities 913, and device drivers 914, and the like. These various software products are among those that will be updated by the service provider computer 102.

In accordance with the present invention, the client computer 101 executes the client application 104 in memory 900. The client application 104 is comprised of a number of executable code portions and data files. These include a security module 901, a communications module 903, a payment module 905, a registration module 904, an advertising and news module 906, a system analyzer 907, a recovery module 908, an install monitor 910, and data defining the current state 911 of the application. The client application 104 further maintains in a private area of the computer storage archive files 909 that archive the state of the client computer 101 prior to each update installation. The client application 104 may be provided to the client computer 101 on a computer readable media, such as a CD-ROM, diskette, 8mm tape, or by electronic communication over the network 106, for installation and execution thereon.

Analysis of Installed Software Products and Determination of Applicable Updates

In the preferred embodiment, the analysis 204 is preferably performed by the client application 104 on the client computer 101. This reduces the network bandwidth required, and the potentially unreliability of non-stateless remote procedure call implementations by having the service provider computer 102 perform the analysis. It further increases the number of simultaneous users of the service provider computer 102. The analyze process is performed by the system analyzer 907 module of the client application 104.

In this embodiment then, the client computer 101 stores a local copy of the method table 801 and the product locator table 803 and uses these local copies to perform the analysis.

Referring now to Figure 10 there is shown the process of the system analyser 907 for analyzing 204 the client computer 101 to determine the list of installed software products.

The system analyzer 907 first synchronizes 1001 the method table 801 and the product locator table 803 in the client computer 101 with the current versions held by the service provider computer 102. Preferably each table is replaced in its entirety; this is likely to be faster than comparing individual entries and updating only those that are out of date. The synchronization may be mandatory or conditioned by version on client computer 101 being older than the version on the service provider computer 102, as indicated by stored timestamp of last time the update table 709 in the service provider computer 102 was updated.

Once the tables are synchronized, the system analyzer 907 can operate locally, for improved efficiency. The system analyzer 907 traverses the entire method table 801, and invokes 1003 each scan method 812 to search the Registry and configuration files 915 of the client computer 101. Each scan method 811 outputs a scan_string, as described, specifying some software product installed on the client computer 101.

The system analyzer 907 applies (1005) each of the scan_strings to the product locator table 803. The product locator table 803 receives the scan_string and resolves 1007 the scan_string to determine a product name 815 and a release instruction 816 associated with it. In some cases, the scan_string does not uniquely identify a product name 815, but matches several product names of installed software products. Accordingly, for each matching entry, the system analyzer 907 obtains 1009 a constraint 814 from the product locator table 803, and resolves 1009 the constraint to determine whether product on the client computer 101 is in fact the product listed in the entry. The constraint 814 of one of the entries will be satisfied and uniquely identify the product name.

Once the specific entry with the correct product name is identified, the system analyzer 907 resolves 1011 the release instruction 816 for the entry to obtain the release or version number of the installed software product. The release instruction 816 is preferably an executable procedure that obtains the version number from the named software product, and thus not merely the actual data itself. Using an executable procedure here ensures that the obtained release or version number is actual value for the product.

The result obtained by the system analyzer 907 from the product locator table 803 is a list 1013 of the installed software products on the client computer 101, each product identified by name and the installed version. The system analyzer 907 uses this list to query the service provider computer 102 to determine 205 for which of these products there is an applicable update.

For each installed product (1002) the system analyzer 907 queries the service provider computer 102 to resolve 1004 the name 815 and release number 818 of the product and determine if there a current update 821 for the product. This may be done by passing in the entire list as name, value pairs, or individually quarrying the service provider computer 102. In either cases, the service provider computer 102 determines if there is an applicable update for a software product by comparing the product name 815 and release information 818 to the product table 805, and obtaining the information in the latest update field 821. If there is an update available, in that the release information in the table indicates a version later than the version that is installed on the client computer 101, then the service provider computer 102 returns 1006 a handle the update ID 819 to the system analyzer 907. If the release of the software product installed on the client computer 101 is the most recent version, then the service provider computer 102 checks the next entry. This process continues until all of the installed software products are checked.

Selection of Software Updates

Once all of the installed software products have been reviewed against the product table 805, the system analyzer 907 will have a list 1007 of the applicable software updates, as those products for which it received an update ID 819 from the service provider computer 102. The system analyzer 907 can then display 206 the list to the user. An exemplary user interface is described above with respect to Figure 4.

The system analyzer 907 can further display 207 additional information for a software update, as illustrated in Figure 5, by querying the service provider computer 102 with the update ID 819 of a particular product to resolve 1008 the update ID 819 on the update table 807 and return information, such as cost, description, and the like.

Installation of Software Updates and the Install Monitor

The user selects one or more of the list software updates. For each selected update, system analyzer 907 returns the update ID 819 to the service provider computer 102. The service provider computer 102 resolves the update ID 819 against the update table 807 to obtain the record for this update, including the URL list 823 identifying the location of the relevant update files. This record is returned to the client computer 101. The client computer 101 accesses the identified URL(s) and downloads the software update files, typically from the software vendor computer 103, though downloads may be from mirror sites, or the like. The client computer 101 further downloads (from the received URLs) any additional installation files, such as installation executables, and scripts. The client computer 101 also verifies that the software update files are not corrupted.

In a preferred embodiment, the client computer 101 employs its security module 901 to verify the integrity of the files to make sure that they have not been corrupted.

The software update is then installed 212 by the client application 104 as described, using the format information 825 to determine the particular installation process, and the script 826 to control any custom installation or configuration information.

Installation 212 is monitored by the install monitor 910, which is executed prior to the actual installation. The install monitor 910 documents the state of the client computer 101 prior to installation and the changes made during the installation of a software update. The install monitor 910 operates in the background, and intercepts calls to the file system or other operating system calls that might result in changes to any files in the client computer 101. Depending on the specific call, the install monitor 910 takes action to preserve the state of the file before the change is made.

Figure 11 illustrates a flowchart of the operation of the install monitor 910. The install monitor 910 receives operating system calls and messages from the client application 104. On trapping 1101 an operating system call, the install monitor 910 determines 1103 the type of call. There are three types of calls of interest: calls 1105 that delete a file or directory, calls 1107 that change an existing file by writing to it, and calls 1109 to add new a file or directory. When a file or directory is to deleted, the install monitor 910 first makes 1113 a copy of the existing file or directory to a private area of the client computer's 101 hard disk or other storage device. The install monitor 910 then lets the operating system 917 delete the file or directory, and waits for the next call. When a file is to be changed 1107, the install monitor 910 determines 1115 whether this is the first write to the file. If so, then again, the install monitor 910 copies 1119 the file to the private area. If the file has been already changed during the installation, there is no need to copy it again. These copy operations 1113, 1119 preserve the configuration of the client computer 101 prior to the installation. Finally, if a new file or directory is to be added 1109, the install monitor 910 stores 1117 the pathname of the new file or directory. This allows the new file or directory to be later deleted during an undo of the installation. For all other types 1111 of operating system calls, the install monitor 910 passes them through without action.

The install monitor 910 waits for installation process 212 to complete, preferably indicated by a message from the client application 104. At this point the complete prior configuration of the client computer 101 is known from the copied files and pathname information. These files and information are compressed 1121 into an archive file 909 and saved on the client computer 101, along with information identifying the software product installation to which it belongs. This identifying information allows the recovery module 908 to retrieve the archived information and restore the configuration of the client computer 101.

Other Service Provider Software Architecture

Referring again to Figure 7, the remaining modules of the service provider computer 102 are now explained.

Communication

The communications module 703 provides for the establishment, maintenance and termination of network connections between the service provider computer 102 and either the software vendor computers 103 or the client computers 101. The communications module 703 supports the FTP and HTTP protocols for sending and receiving data over the Internet and the World Wide Web. The communications module 703 generally maintains and establishes separate streams for each connection it maintains. Preferably, the service provider computer 102 supports a large number of connections, possibly several hundred or thousands, at a time. In the event the customer base is so large that an even larger number of simultaneous connections may be required, multiple servers with mirror images of the update database 709 may be used. The communications module 703 also handles login and logout in a conventional manner, though these functions may be incorporated into the security module 701, below.

Security

The security module 701 handles the authentication of the user as an authorized user of the service provider com-

puter 102. The security module 701 may be implemented with conventional authentication mechanisms based on digital signatures, such as public key systems supporting digital signatures, certificates and the like. Suitable security mechanisms include VeriSign Inc.'s Digital ID Center, which incorporates the login and logout functions from the communications module 703.

Additionally, the security module 701 provides for verification of the integrity of software updates that are downloaded from software vendor computers 103 to ensure that such updates have not been altered or infected by computer viruses or other unauthorized modifications. This module may be used, for example, to compute a checksum of the updates and the checksum may be stored in the update database 709. The checksum may be a simple one, or a cryptographically secure one such as any of the Message Digest (MD) algorithms proposed by Professor Ronald Rivest and commonly available in programming API's such as Microsoft's Cryptographic API standard. Whenever an update is later downloaded to a client computer 101 from a software vendor computer 103, the checksum of the update may be computed and compared against the one stored in the update database 709. If the two match, it may reasonably be inferred that the software update was downloaded to the client computer 101 correctly. The security module 701 may also be used to scan for viruses in the software updates stored on the various software vendor computers 103.

Payment

The payment module 705 handles payment by the end user to the service provider for the service of providing software updates. The service provider computer 102 maintains a database of its users. This database may be the user profile database 711 or other databases. Each user is charged a service fee for using the service provider computer 102 to download software updates. The fee may be based on a variety of different schedules, such as connection time, number of software updates purchased, annual or monthly subscription fee, or a combination of any of these or other pricing formulas. However charged, the payment module 705 tracks the user's usage of the service, for example, total the connection time, and maintains a count of the number of software updates downloaded, until the user logs out of the service provider computer 102. Payment is then charged to the user's credit card, which was previously supplied by the user during registration. Suitable implementations of the payment module 705 may be created in conformance with the Secure Electronic Transaction specification of Mastercard and Visa.

A user's subscription to the service may be enforced by the payment module 705 in various ways. One example of an algorithm to enforce term subscription is as follows:

The user logs in from the client computer 101 to the service provider computer 102. The payment module 705 determines if the user's account is current, and if so, accepts the connection to the client computer 101. If the user's account is about to expire, for example, within 30 days, or has expired, the payment module 705 prompts the user to renew the subscription. If the user agrees, the subscription fee is charged to the user's credit card account, and the connection to the client computer 101 is established, allowing the user to use the service as described. If the users refuses to renew, the connection is refused.

Fees may also be charged on a per-transaction basis. In this scenario, the fees may be attached to selected transactions. Once example of an algorithm to enforce per-transaction fees is as follows:

The client application 104 requests, for a software product to be updated, a transaction permission from the service provider computer 102. The payment module 705 determines from the update database 705 a specific fee for the transaction, and returns this information, along with a permission, to the client application 104. The client application 104 displays the fee to the user, who either confirms the transaction or cancels the software update. If the transaction is confirmed, the client application 104 performs the installation process. The payment module 705 is notified if the transaction and installation is successful, and then adds the transaction fee to a running total of fees for the current session. When the user's session is complete, the running total of transaction fees is charged to the user's credit card, and the charges provided to the client application 104 which displays them to the user.

In cases where an update is going to be undone by the recovery module 908, the transaction fees should to be credited back to the user's credit card account. Here, the client application 104 informs the service provider computer 102 that a software update is to be undone, providing the update ID 819 of the software update The payment module 705 uses the update ID 819 to determine the transaction fee (cost 824) to be credited. This amount is passed back to the client application 104 and displayed to the user. The software update is removed by the recovery module 908, and the payment module 705 is notified of the successful removal. The payment module 705 then subtracts the transaction fee from any current running total of fees. At the close of the session, the payment module 705 either charges or credits the user's credit card account, as appropriate.

Database Modification

The database modification tools 707 provide for the maintenance and updating of the update database 709 to include new software updates from various software vendors. The tools 707 provide for the addition of new entries, and the deletion or alternation of existing entries in any of the tables of the update database 709.

Of the various tables, the update table 807, which contains the information about the current updates for the software products, and the product table 805, which identifies the various software products for which their are updates, are the most frequently modified.

As new software updates become available, either the service provider or the software vendors access the database modification tools 707 to update the database. This is preferably done by completing forms that capture the information used in the tables of the database. Figure 13 illustrates a sample form for specifying new update information, or changing existing update information. The form 1300 includes fields for providing the remark 1301 used in describing the update, a URL 1303 for the information on the software update, version information 1305, software products 1307 affected by the update, the type of update 1309, known incompatibilities 1311, filters for locating prior versions of the software product to be updated based on version information 1313, date information 1315, and Registry information 1317 (for identifying the software product in the Registry files of the 915 of the client computer 101). In addition, the file format 1321 of the update is specified along with a URL 1319 for the network location of software update itself. Finally, the installation procedures 1323 are specified for use in an installation script 826. This information readily processed in a conventional manner and updated to the appropriate tables of the update database 709.

In order to be supported by the update service of the service provider, software products and the updates to the software products have to be registered in the update database 709.

Registering a software product has the goal of specifying sufficient information to identify a product and its version if the product has been installed on a given client computer 101. Figure 17 illustrates a form for registering a software product into the update database 709 for the first time. The registration form 1700 contains fields for the software vendor's company name 1701, software product name 1703, product type 1705, a method 1707 to identify the software product on the client computer 101, a unique file name 1707or character string identifying the product, methods 1709 for verifying version information, file dates 1711, and directories 1713 on the client computer 101.

The product type 1705 can be a device driver, an application, a plug in (a product which extends the capabilities of another product such as an Internet browser) or an operating system file.

The method 1707 to identify the software product preferably specifies a unique file name or a character string and the location of the file name or string. For example, on the Windows 95 operating system, the name of a sound driver is specified in the Registry location \HKEY_LOCAL_MACHINE\System\CurrentControlSet\control\MediaResources\midi. In this case, the filename of the driver is found in this Registry location. A software product could also be identified by the presence of unique directory names. As noted, in some instances, product names are not unique.

The version of the software product that is installed on a client computer 101 may be obtained in one of several ways. It may be the version number, the last modification time-stamp of a file, or it may be specified explicitly in the Registry. The information provided in the registration form is processed after submission and added to the appropriate tables of the update database 709.

Software updates may be identified for inclusion in the update database 709 by the service provider periodically searching the Internet to identify software vendors providing updates of software products. Most software vendors will maintain Internet sites that indicate the presence of new software updates. For each identified software vendor, the service provider downloads the software updates to the updates database 709. A file format of the software update is determined, and an installation process specified according to the file format of the software update. Finally, the service provider creates an entry in the update database 709 including the URL or network location of the software vendor's computer system 103 storing the software update, the file format of the software update, and a specification of the installation process.

Alternatively, software vendors who contract with the service provider may provide the information about their software products and software updates, e.g. name, file format, and so forth, directly to the service provider, or to the update database 709.

However provided to the update database 709, registering an update consists of specifying the properties of the software update and the software products and their versions to which the software update is applicable. The properties of the software update preferably include the new version number 820 that results if the software update is applied to the product, the format 825 of the software update - zip file, self-extracting archive, and the like, and the installation steps (script 826) required to install the software update on the client computer 101. The product versions to which the software update is applicable are specified as the products themselves are specified earlier in this section. Also, a URL to a brief description and a full description of the software update - the problems it fixes and features it might add -is preferably included, or the information may be directly stored.

As each new update becomes available, a new update entry is created.

Either the software vendor or the service provider specifies the product and the software update database entries in conformance with the properties of the software update.

User Profile Database

The user profile database 711 maintains a profile for each user containing information about which products the user has shown an interest, for example by requesting notification about software updates for specific products, or about new software products. This information is then used to deliver notifications about new updates available for these products to the user, for example by email, or other electronic communications mechanisms. This optional feature of the service provider computer 102 further enhances the value of the service to the user, ensuring timely notification of the availability of software updates and new software products.

In this regard, one alternate embodiment of the present invention is the use of email to notify users about new software update information, and new software products for which the user has expressed an interest. Specifically, when a new software update or software product is available, the service provider computer 102 sends an email to those users who have requested notification by email. The email contains information about the software update, and may include the record from the update table 807 about the software update, including the URL data 823 used to access the software update files. The client application 104 would then read the update information, and verify that the software update is indeed applicable to the client computer 101, and that the client computer 101 satisfies any conditions for installation. If the software updates are approved by the user, the client application 104 downloads the software update, verifies its integrity, and installs the software update directly, without having to login 201 to the service provider computer 102, and analyze 204 the software products installed on the client computer 101. In the case of notifications about new software products in which the user had expressed interest, the client application 104 would verify that the user is still interested in the software product and proceed to purchase, download and install it.

As a further enhancement of the e-mall notification embodiment, the email sent by the service provider computer 102 includes a specification of conditions a client computer 101 must satisfy for the software update or software product to be installed. This information is essentially the same as that used by the client application 104 to determine the relevant software updates for the client computer 101. For example, this information includes, for a software update, the older versions of the software product to which it is applicable. This additional information in the email notification is used by the client application 104, for example, to ensure that the software update is used only once by the user, and can be repeatedly applied.

The user profile database 711 generally stores information descriptive of each user. This information may include the user ID, password, digital signature, credit card numbers and the like, for use by the security 701, communications 703, and payment 705 modules. Figure 14 specifies one exemplary schema of the user profile database 711. In a user table 1400, each user is identified by user ID 1401, name 1403, email address 1405, the start date 1407 of their subscription to the service, the end or termination date 1409 of the subscription, credit card information 1411 such as number, issuer and expiration date, a user selected password 1413, and a public key 1415 or other authentication token. As illustrated in Figure 3, the user has the option 309 of requesting notification by email of such software updates. The user table 1400 thus also includes a flag 1416 indicating whether the user so desires to be notified by email. The user table 1400 is keyed by the user ID 1401 to a notification table 1417 that associates the user with selected product names 1419 and their current version 1421. When a software vendor or the service provider updates the update database 709 with information for a new software update, the notification table 1417 may be scanned to identify those users by user ID 1401 to notify about the update. The email flag 1416 for a user is checked, and if true, the user's email address 1405 is obtained from the user table 1400 and the user notified by email with information identifying the new software update.

Activity Log

The service provider computer 102 may be used to log all activities it performs with respect to the service in the activity log 718. Of particular interest are the activities the computer performs in response to user requests for software updates and the like. An illustrative format for the activity log 718 is shown in Table 2.

Table 2

Activity Log 718					
Transaction Id	Activity Type	Date-Time	User Id	Parameters	Response
00000001	Login	031296 093540	20198312	password	Success
00000002	GetMethods DB	031296 093606	20198312	last version	Methods DB *or* Up-to-date
00000003	GetProducts Locator DB	031296 093649	20198312	last version	Products-Locator DB *or* Up-to-date
00000004	Query Product DB	031296 093723	20198312	Sound Blaster16, 2.0	sb-2.02
	Query Product DB	031296 093727	20198312	Myst 1.0	Up-to-date
00000005	GetUpdate Entry	031296 093751	20198312	sb16-2.02	Update Entry
00000006	Download Done	031296 093807	20198312	sb16-2.02	Success
00000007	Installed Update	031296 094532	20198312	sb16-2.02	Success
00000008	Logout	031296 094730	20198312	-	Success

In this example, the user logged in on March 12, 1996 at 09:35:40 a.m., synchronized their method table 801 and product locator table 803, queried if software updates for SoundBlaster16 2.0 and Myst 1.0 newer than these product's last version were available. The responses indicate that Myst 1.0 was update to date, but the current version of SoundBlaster16 was version 2.02. The user then obtained the update entry for the new version of SoundBlaster16 describing the software update, downloaded the software update, installed it, and logged out.

Activity types not represented in the example above include Undo of Updates by the recovery module 908, registering for service, and registering for notification for updates to specific products.

In this example, the activities of a single user are represented in the activity log. In an actual system, the activities of several different users would be interspersed in the activity log.

Reporting Tools

The reporting tools 713 provide support for querying the update database 709, the user profile database 711 and the activity log 718. The queries may be about the software products and updates, about the correlation between the types of software updates accessed by various users, and about aggregate data. The databases 709, 711 and the activity log 718 together have the potential to provide precise descriptions of the software product profiles of the users. For example, statistical information may be retrieved indicating the number of users of one product, such as SoundBlaster16, who also own a second product, such as Myst. This information may be collected and analyzed without necessarily violating the privacy of the individual users.

URL Monitor

The URL monitor 715 compiles the list of URLs in the update database 709 and verifies on a periodic basis whether they have changed. This is done to ensure that the URL information for the software updates is always valid. Figure 12 illustrates a flowchart of the URL monitor 715. The URL monitor 715 traverses 1201 each entry in the update table 807.

This may be done simply in serial order, or by more complex approaches, such as oldest entries first, or some other fashion. For each entry, the URL monitor 715 obtains 1203 the URL entries in the URL list 823, each entry as noted above having a timestamp. The URL monitor 715 links 1205 to the URL in an attempt to connect to the identified site or file via the Internet.

The attempted link may fail, and may be repeated some number of times in order to confirm that the URL is actually absent or otherwise incorrect, as opposed to merely a failure of the network service provider or the like. Once it is determined 1207 that the URL is not present, the URL is marked 1209 in the update table 807 as being invalid.

If the URL is present, then the timestamp of the URL at the host site is checked, typically by checking the timestamp of the file associated with the URL, or the timestamp of the file that includes the URL, or whichever is later. If the timestamp at the host is newer than the timestamp held in the update table 807, then it is possible that the underlying file has been changed, and the URL is no longer valid. Again, the URL is marked 1209 as being invalid. If the timestamp of the host is not newer, then the URL monitor 715 continues with the next URL in the URL list 823. Once all of the URLs in the update table 807 (or the desired number of old ones) have been processed, then the URL monitor 715 notifies 1213 the system administrator of the potentially invalid URLs. The system administrator can then verify the URLs and update them if necessary, resetting the valid flag as the URLs are updated.

Advertising & Information Database

The access that the service provider computer 102 has to the software profile of the client computers 101 lends itself to sending information, advertisements, and other promotional material that would be appropriate to each specific user, based on the software installed on the user's computer. Basing information delivery on the installed software products increases the saliency of the information since the user has already manifested an interest in the products. Thus, advertising or promotional information that is derived from or associated with such software products is most likely to be of interest to the user. The service provider computer 102 associates software products with advertising information, and enables this advertising information to be periodically delivered to the user.

Furthermore, the nature of downloading and installing software updates is inherently time-consuming; the risks that users perceive in updating usually would mean that they would seldom perform the updates on unattended computers. These factors create an opportunity to the service provider to direct targeted advertisements at the user at appropriate moments when the user runs the client application 104 to update their software, at which time they are present at their computer but not engaged in other activities. The advertisements themselves may be about for-fee software updates (upgrades) that the user may be able to purchase from the service provider or other third parties. Delivery of advertising information during the update process 212 is on the client computer 101 by the advertising/news module 906.

The advertising and information database 717 accordingly associates software products with advertising and promotional information. This association may be made in a number of different ways. One mechanism of association is categorizing software products and advertisements. Figure 15 illustrates an exemplary schema for the advertising and information database 717 for associating advertising information and software products.

The ad table 1500 includes for each advertisement an ad number 1501, a URL 1503 to the advertisement or information item, and a list 1505 of categories for the advertisement, such as "word processing," "desktop publishing," "graphics," "adventure games," "communications," "Internet" and the like. An advertisement or information item may have any number or variety of categories associated with it. The product-category table 1507 lists products names 1511, product IDs 1509, and again, a list 1513 of categories for the product.

If a user has requested updates to a specific installed product, then presumably the user would be interested in advertisements or information for other products that are categorized in the same categories as the installed product. For example, if the user requests an update to an installed copy of Myst 1.0, then this product name is matched against the product name 1511 in the product-category table 1507, and the categories 1513 for it, such as "interactive game," are retrieved. The categories 1505 in the category list 1505 of the ad table 1500 are matched against this category, and the URLs 1503 for matching entries retrieved and accessed, with the information being delivered to the user by the client application 104. The information is preferably presented on the client computer 101 during the installation process 208-214. If there are many matches, then a weighting may be applied to select only those advertisements that match a certain percentage, or number, of categories of the installed products. Other selection criteria may also be applied. The schema of Figure 15 is merely illustrative, and implementations other categorization may be used to associate advertising information with software products for delivery to users having such products installed on their computers.

Client Application Software Architecture

Referring again to Figure 9, the remaining modules of the client application 104 are now explained.

Communication

The communications module 903 provides complementary functions to the communications module 703 of the service provider computer 102, including establishing and terminating connection streams, login and logout functions, FTP functions, and HTTP protocol compliance. All of these functions may implemented in a conventional manner.

Security

The security module 901 provides an interface to the security module 701 of the service provider computer 102, for authentication of the user password, digital signatures, certificates, or the like. User passwords or other authentication information are assigned to the user in a conventional manner. The security module 901 may store the authentication information, or the user may be required to manually input the authentication information during login.

Payment

The payment module 905 provides an interface to the payment module 705 of the service provider computer 102 to effect payment for use of the update service. Payment schedules may vary as described above. Preferably payment is made by credit card authorization. Given one or more payment schedules for use of the service, such as per update, periodic fees, or the like, the payment module may be implemented in a conventional manner.

Registration

The registration module 904 is used to register new users to the service provider computer 102. A sample user interface for the registration module 904 is shown in Figure 3.

The registration module 904 obtained the user's name, address, credit card information, and a user-selected password. The password is entered by the user twice and the two entries matched to ensure that the user did not mistype the password unintentionally. This information is stored in the current state 911 data. The registration module 904 also sends this information to the service provider computer 102. There the information is verified and a unique registration number assigned to the user and the number is returned to the client application 104, where the registration module 904 displays the number to the user, and stores the number internally in the current state 911 data.

Advertising & News

The advertising and news module 906 provides customized information to each user of the service based on their prior interests in various software products and updates, as monitored and stored in the user profile database. The advertising and news module 906 interfaces with the advertising database 717 of the service provider computer 102 to deliver advertising and promotional information the user based on the installed software products on the user's computer 101.

The advertising and news module 906 provides information in various different modes. In one mode, the advertising and news module 906 obtains ads from the advertising database 717 on a periodic basis, such as once every several hours, according to the installed software products on the client computer 101, as described above, and caches them locally. If an ad (here including other types of information or promotional data) is already present in the cache, it is marked as new, otherwise, the URL of the ad (as determined from the database 717) is accessed, and the ad saved in the cached. Ads not marked as new are purged.

In a second, complementary mode, the advertising and news module 906 then selects ads from the cache and displays them to the user for a predetermined duration when no other user activity is occurring, such as during the installation process described above, or during an undo operation by the recovery module 908.

Current State

The current state 911 is a data store of data describing the present operation of the client application 104, including for example, user specific information, such as name, address, credit card number, registration or serial number, and which updates have been downloaded and which have been installed. The registration number is used each time the user logs in to the service provider computer 101. The information about which updates have been downloaded and installed is used to provide the undo capability of the recovery module 908.

Recovery

The recovery module 908 provides for undoing, or de-installing previously installed software updates using the

archive files 909.

Recovery is an action initiated by the user when he or she is dissatisfied with a software update. When initiated, the effects of a software update previously installed are reversed. The ability of the recovery module 908 to perform the recovery is based on the presence of the archive files 909 created by the install monitor 910 when the software update was first installed. The archive files 909 contain a copy of each file which was deleted or modified during installation along with its original pathname and a list of pathnames of files added during the installation. The archives 909 are preferably kept in a compressed format for space efficiency. Generally, given a specific software update to remove, the recovery module 908 reads the archive file 909 associated with that software update, restores the deleted or modified files to their directories, and deletes the added files or directories.

Figure 16 illustrates one embodiment of the operation of the recovery module 908. The recovery module 908 receives, as shown in Figure 6, an input of the name of the software update to be removed. This name is associated in the current state information 911 with the particular archive file 909 for that installation. The recovery module 908 doses 1601 all executing applications. Using the name of the software update, or other identifying indicia, the recovery module 908 obtains the archive file 909 associated with the update, and uncompresses 1602 it. For each file that is stored in the archive in compressed form, representing a file that the was deleted, the recovery module 908 copies 1603 that file to its original location in the client computer 101. For each file or directory that is listed as being new, the recovery module 908 deletes 1604 that file or directory. Finally, the recovery module 908 reboots 1605 the client computer 101.

In summary, the present invention enables a useful mechanism for providing updates of various software products from diverse software vendors to a plurality of users, each user having different ones of the software products installed on their computers. The system of the present invention enables the software updates to be continually maintained and verified for correctness, while alleviating both users and software vendors of a substantial burden is communicating with each. The system enables any software vendor to provide software updates to the service provider, ensuring that subscribing users have the software update on a timely basis. Likewise, subscribing users are ensured that they are notified about software updates for all of the software products installed on their computers, without having to individually search out information for each such product. In addition, the present invention enables advertising and other information to be targeted to users based on their interests and preferences and expressed in the software products installed on their computers.

Claims

1. A computer implemented method of providing software updates for a plurality of software products from a plurality of software vendors to a least one user computer, comprising:

 maintaining a database of information about software updates for a plurality of software products from a plurality of software vendors, the information including for each software update a network location of a computer system storing the software update;
 downloading at least a portion of the database to the user computer;
 determining on the user computer the software updates that are relevant to the user computer according to the software products installed on the user computer;
 receiving a user selection of at least one applicable software update to install;
 downloading a selected software update from the network location specified in the database; and,
 installing the downloaded software update on the user computer.

2. The computer implemented method of claim 1, wherein:

 the database includes for each software update a specification of an installation process for installing the software update on the user computer; and,
 installing the software updates includes installing the software update according to the installation process specified for the software update in the database.

3. The computer implemented method of claim 1 or 2, wherein the installation process is specified in accordance with a file format of the software update.

4. The computer implemented method of one of the preceding claims, further comprising:

 providing on each user computer a client application of the database, and for each user computer, periodically activating the client application to:

 connect the user computer to the database;

automatically download said portion of the database to the user computer;
automatically determine the software updates that are relevant to the user computer according to the software products installed on the user computer;
receive said user selection of at least one applicable software update for installation;
download said selected software update from the network location specified in the database; and,
install the downloaded software update on the user computer.

5. The method of one of the preceding claims wherein maintaining a database of information comprises:

receiving from a software vendor information describing a software update of a software product, including a file format of the software update, a network location of the computer system storing the software update, and an installation process for installing the software update; and,
creating an entry in the database storing the received information.

6. The method of claim 5, wherein installing the software update includes:

determining from the database the file format of the software update; and,
executing the specified installation process on the downloaded software update.

7. The computer implemented method of one of the preceding claims further comprising:

periodically searching the Internet to identify software vendors providing updates of software products.

8. The method of one of claims 4 - 7 wherein:

the database includes for a plurality of software products of diverse software vendors promotional information for the software products in association with a network location for the promotional information;
wherein the client application automatically:

determines promotional information relevant to the user computer software products installed on the user computer;
downloads to the user computer the relevant promotional information according to the network location associated with the such information.

9. The method of one of the preceding claims wherein downloading a software update from the network location specified in the database comprises:

intermediating between a computer system controlled by the software vendor providing the software update and the user computer to enable the user to provide electronic payment for the software update to the software vendor.

10. The method of one of the preceding claims wherein downloading a software update from the network location specified in the database comprises:

verifying that the downloaded software update is authentic and not corrupted.

11. The method of one of the preceding claims, wherein installing a software update comprises:

monitoring any changes made to the user computer during installation;
for each change to the user computer, storing data descriptive of the change;
making the changes to install the software update; and,
archiving the stored data as relating to the software update.

12. The method of claim 11, further comprising:

receiving a user selection of an installed software update to be removed from the user computer;
retrieving archive files assocated with the installed software update;
restoring from the stored data of the archive files the user computer to a state prior to the installation of the software update by undoing the changes described in the stored data.

13. The method of one of the preceding claims, wherein installing the downloaded software update comprises:

determining whether a file existing in the user computer is to be deleted or altered, or a new filed added to the user computer;
responsive to a file being deleted, saving a copy of the file and then deleting the file;
responsive to altering file that has not been previously altered, saving a copy of the file, and then altering the file;
responsive to a new file to be added to the user computer, storing a pathname of the file to be added, and then adding the file; and,
archiving the copied files and stored pathnames in at least one archive file that is associated with the installed software update.

14. The method of one of the preceding claims, further comprising:

receiving a user selection of an installed software update to be removed from the user computer;
retrieving archive files associated with the installed software update;
deleting files added during installation according to stored pathnames in the archive files; and,
restoring files deleted or altered during installation from the copied files.

15. The method of one of the preceding claims, wherein determining the software updates that are applicable to a user computer comprises:

identifying each software product installed in the user computer;
determining a version of each installed software product;
identifying from the database those software updates for the installed software products having a version that is later than the version of the software product installed on the user computer.

16. The method of claim 15, wherein identifying each software product comprises:

uniquely identifying two software products having a same name, by using a constraint to distinguish between the software products.

17. A computer implemented method of providing software updates for a plurality of software products from a plurality of software vendors to a least one user computer, comprising:

maintaining a first database of information about software updates for a plurality of software products from a plurality of software vendors, the information including for each software update a network location of a computer system storing the software update;
maintaining a second database of user information for a plurality of users, including for each user, information identifying at least one software product of interest to the user;
notifying at least one user by an electronic communication of the availability of a software update to a software product of interest to the user, the electronic communication including a network location of the software update;
receiving from the user an approval to install the software update;
downloading the approved software update from the network location specified in the electronic communication; and,
installing the downloaded software update on the user computer.

18. A computer implemented method of providing software products from a plurality of software vendors to a least one user computer, comprising: maintaining a first database of information about software products

from a plurality of software vendors, the information including for each software product a network location of a computer system storing the software product;
maintaining a second database of user information for a plurality of users, including for each user, information identifying at least one software product of interest to the user;
notifying at least one user by an electronic communication of the availability of a software product of interest to the user, the electronic communication including a network location of the software product;
receiving from the user an approval to install the software product;
downloading the approved software product from the network location specified in the electronic communica-

tion; and,
installing the downloaded software product on the user computer.

19. A computer readable memory on a user computer including a computer program for configuring and controlling a processor of the user computer to:

connect with a database of information about software updates for a plurality of software products from a plurality of software vendors, the information including for each software update a network location of a computer system storing the software update;
automatically download at least a portion of the database to the user computer;
automatically determine the software updates that are applicable to the user computer according to software products installed on the user computer;
receive a user selection of at least one applicable software update for installation;
download a selected software update from the network location specified in the database; and,
install the downloaded software update on the user computer.

20. The computer readable memory of claim 19, wherein:

the database includes for each software update a specification of an installation process for installing the software update on the user computer; and,
the computer program stored on the computer readable memory further controlls the process to install the software update according to the installation process specified for the software update in the database.

21. The computer readable memory of claim 20, wherein the installation process is specified in accordance with a file format of the software update.

22. A computer implemented method of providing advertising information to at least one user computer, comprising:

maintaining a database of advertising information for a plurality of software products of diverse software vendors in association with a network location for the promotional information;
determining promotional information relevant to a user computer according to software products installed on the user computer;
downloading to the user computer the relevant promotional information according to the network location associated with the such promotional information.

23. A computer implemented method of removing an installed software update of a software product from a computer, comprising:

monitoring any changes made to the computer during installation of the software update of the software product;
for each change to the computer, storing data descriptive of the change;
archiving the stored data as relating to the software update;
receiving a user selection of an installed software update to be removed from the computer;
retrieving archive files assocated with the installed software update;
restoring from the stored data of the archive files the computer to a state prior to the installation of the software update by undoing the changes described in the stored data.

24. The method of claim 23, wherein storing data descriptive of the change comprises:

determining whether a file existing in the computer is to be deleted or altered, or a new filed added to the computer;
responsive to a file being deleted, saving a copy of the file and then deleting the file;
responsive to altering file that has not been previously altered, saving a copy of the file, and then altering the file;
responsive to a new file to he added to the computer, storing a pathname of the file to be added, and then adding the file; and,
archiving the stored data as relating to the software update comprises:

archiving the copied files and stared pathnames in at least one archive file that is associated with the

installed software update.

25. The computer implemented method of claim 23 or 24, wherein the step of restoring comprises:

restoring files deleted or altered during installation from the copied files stored in the retrieved archive files.

26. The computer implemented method of one of the claims 23 - 25 further comprising:

deleting files added during installation according to stored pathnames in the retrieved archive files.

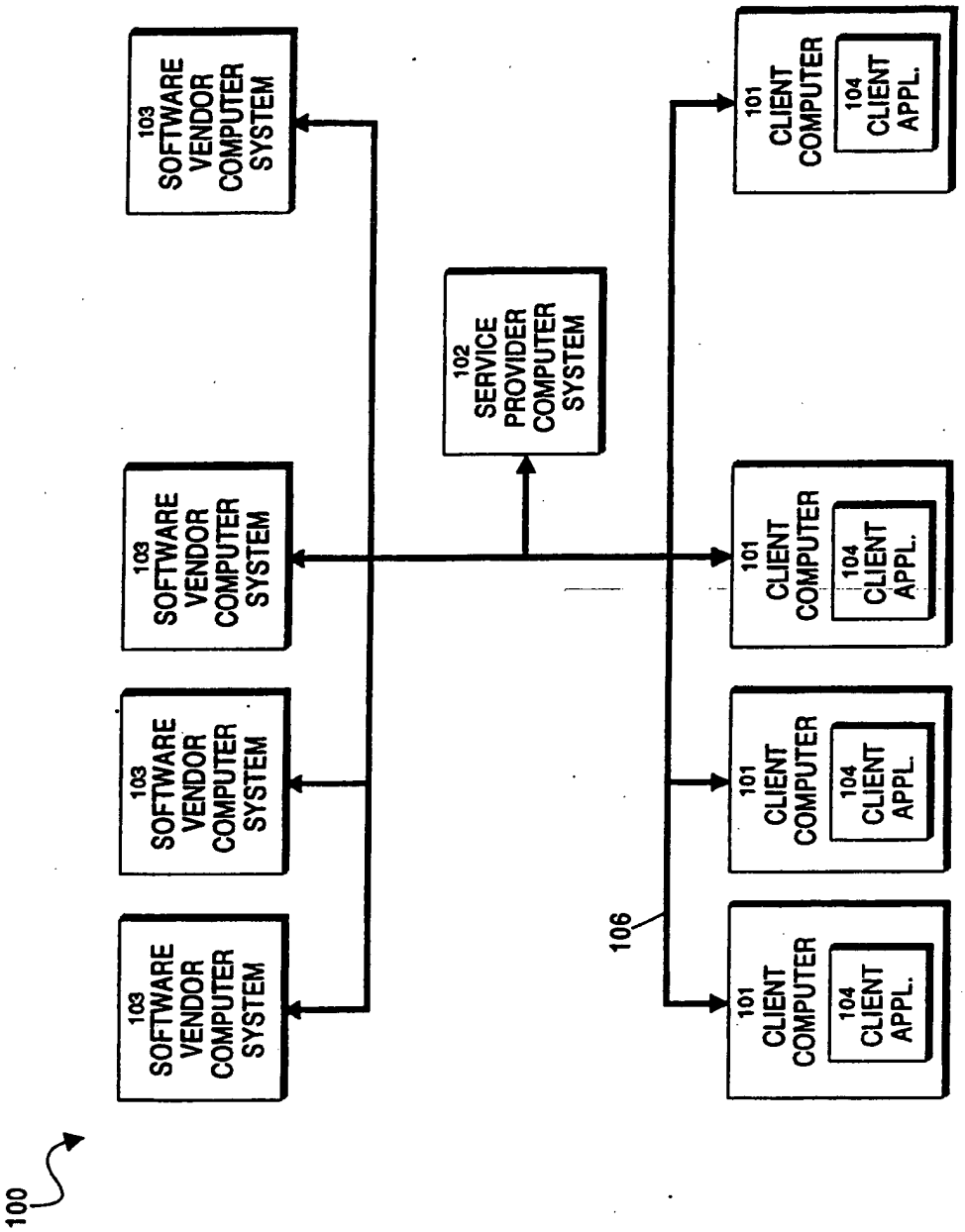

100
103 SOFTWARE VENDOR COMPUTER SYSTEM
103 SOFTWARE VENDOR COMPUTER SYSTEM
103 SOFTWARE VENDOR COMPUTER SYSTEM
103 SOFTWARE VENDOR COMPUTER SYSTEM
102 SERVICE PROVIDER COMPUTER SYSTEM
106
101 CLIENT COMPUTER 104 CLIENT APPL.
101 CLIENT COMPUTER 104 CLIENT APPL.
101 CLIENT COMPUTER 104 CLIENT APPL.
101 CLIENT COMPUTER 104 CLIENT APPL.

FIGURE 1

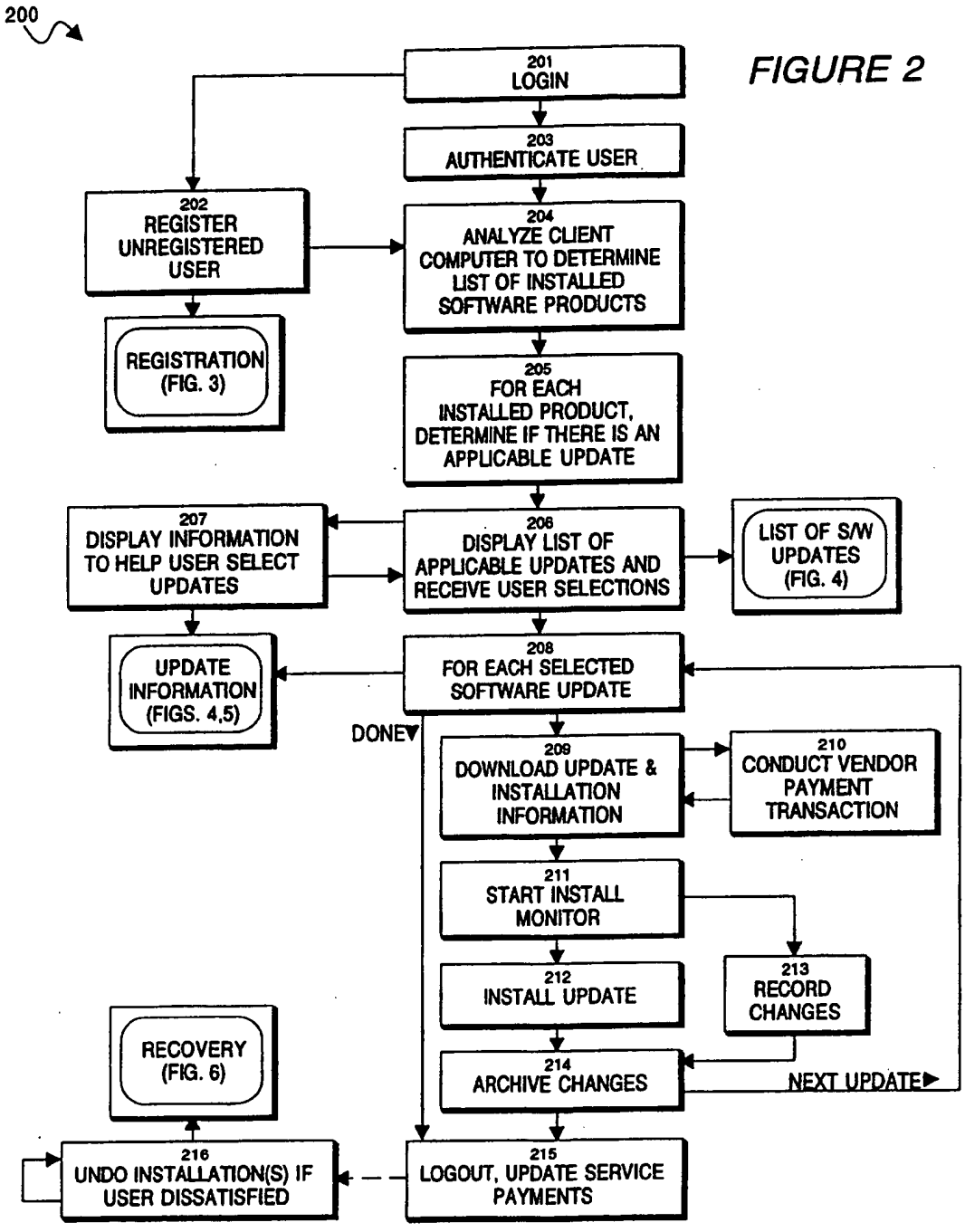

200
201
LOGIN
203
AUTHENTICATE USER
202
REGISTER UNREGISTERED USER
204
ANALYZE CLIENT COMPUTER TO DETERMINE LIST OF INSTALLED SOFTWARE PRODUCTS
REGISTRATION (FIG. 3)
205
FOR EACH INSTALLED PRODUCT, DETERMINE IF THERE IS AN APPLICABLE UPDATE
207
DISPLAY INFORMATION TO HELP USER SELECT UPDATES
206
DISPLAY LIST OF APPLICABLE UPDATES AND RECEIVE USER SELECTIONS
LIST OF S/W UPDATES (FIG. 4)
UPDATE INFORMATION (FIGS. 4,5)
208
FOR EACH SELECTED SOFTWARE UPDATE
DONE
209
DOWNLOAD UPDATE & INSTALLATION INFORMATION
210
CONDUCT VENDOR PAYMENT TRANSACTION
211
START INSTALL MONITOR
212
INSTALL UPDATE
213
RECORD CHANGES
214
ARCHIVE CHANGES
NEXT UPDATE
RECOVERY (FIG. 6)
216
UNDO INSTALLATION(S) IF USER DISSATISFIED
215
LOGOUT, UPDATE SERVICE PAYMENTS

FIGURE 2

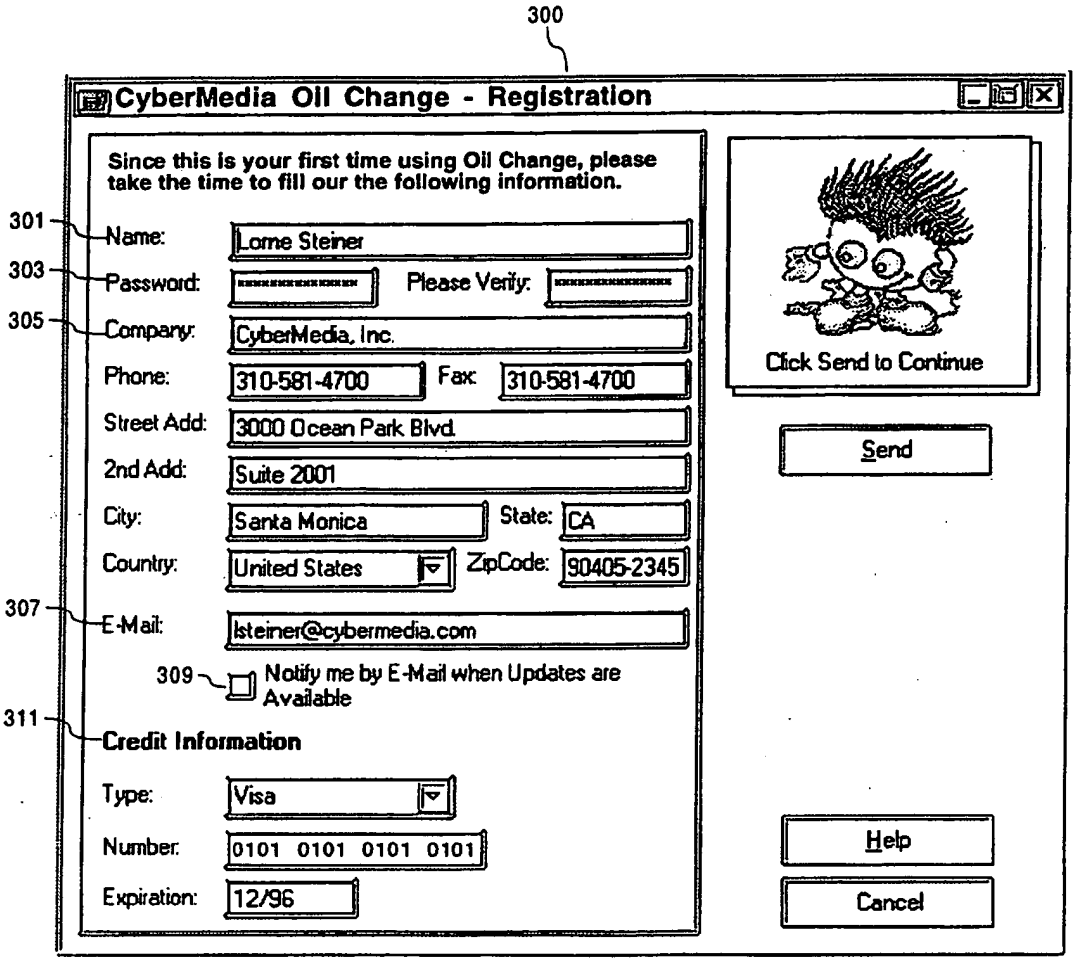

300
CyberMedia Oil Change - Registration
Since this is your first time using Oil Change, please take the time to fill our the following information.
301
Name:
Lorne Steiner
303
Password:
Please Verify:
305
Company:
CyberMedia, Inc.
Phone:
310-581-4700
Fax:
310-581-4700
Street Add:
3000 Ocean Park Blvd.
2nd Add:
Suite 2001
City:
Santa Monica
State:
CA
Country:
United States
ZipCode:
90405-2345
307
E-Mail:
lsteiner@cybermedia.com
309
Notify me by E-Mail when Updates are Available
311
Credit Information
Type:
Visa
Number:
0101 0101 0101 0101
Expiration:
12/96
Click Send to Continue
Send
Help
Cancel

FIGURE 3

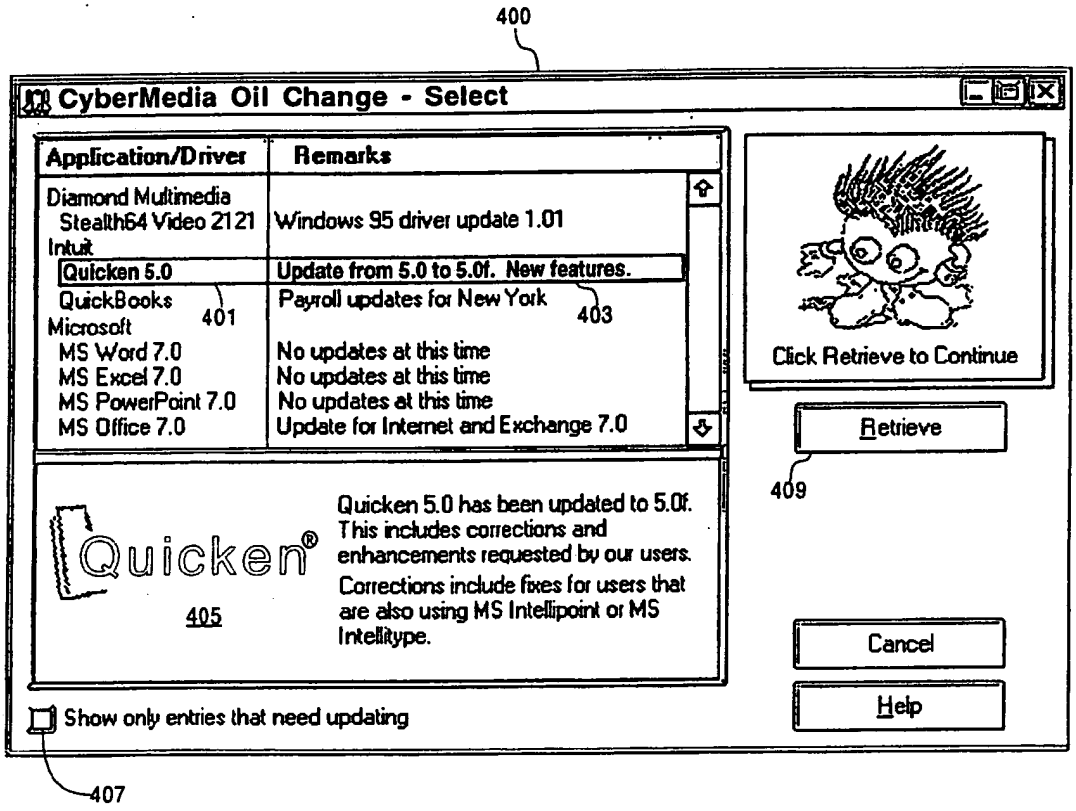

400
CyberMedia Oil Change - Select
Application/Driver
Remarks
Diamond Multimedia
Stealth64 Video 2121
Windows 95 driver update 1.01
Intuit
Quicken 5.0
Update from 5.0 to 5.0f. New features.
QuickBooks
401
Payroll updates for New York
403
Microsoft
MS Word 7.0
No updates at this time
MS Excel 7.0
No updates at this time
MS PowerPoint 7.0
No updates at this time
MS Office 7.0
Update for Internet and Exchange 7.0
Click Retrieve to Continue
Retrieve
409
Quicken®
405
Quicken 5.0 has been updated to 5.0f. This includes corrections and enhancements requested by our users.
Corrections include fixes for users that are also using MS Intellipoint or MS Intellitype.
Cancel
Help
Show only entries that need updating
407

FIGURE 4

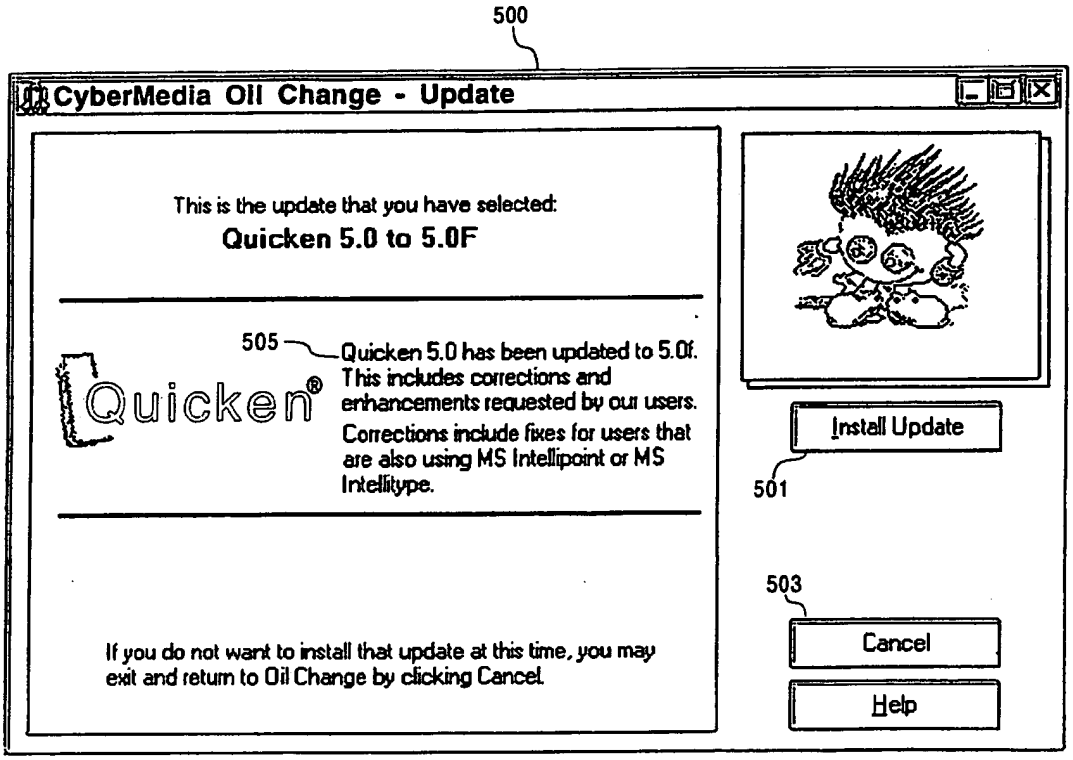

500
CyberMedia Oil Change - Update
This is the update that you have selected:
Quicken 5.0 to 5.0F
505
Quicken®
Quicken 5.0 has been updated to 5.0f. This includes corrections and enhancements requested by our users. Corrections include fixes for users that are also using MS Intellipoint or MS Intellitype.
Install Update
501
503
If you do not want to install that update at this time, you may exit and return to Oil Change by clicking Cancel.
Cancel
Help

FIGURE 5

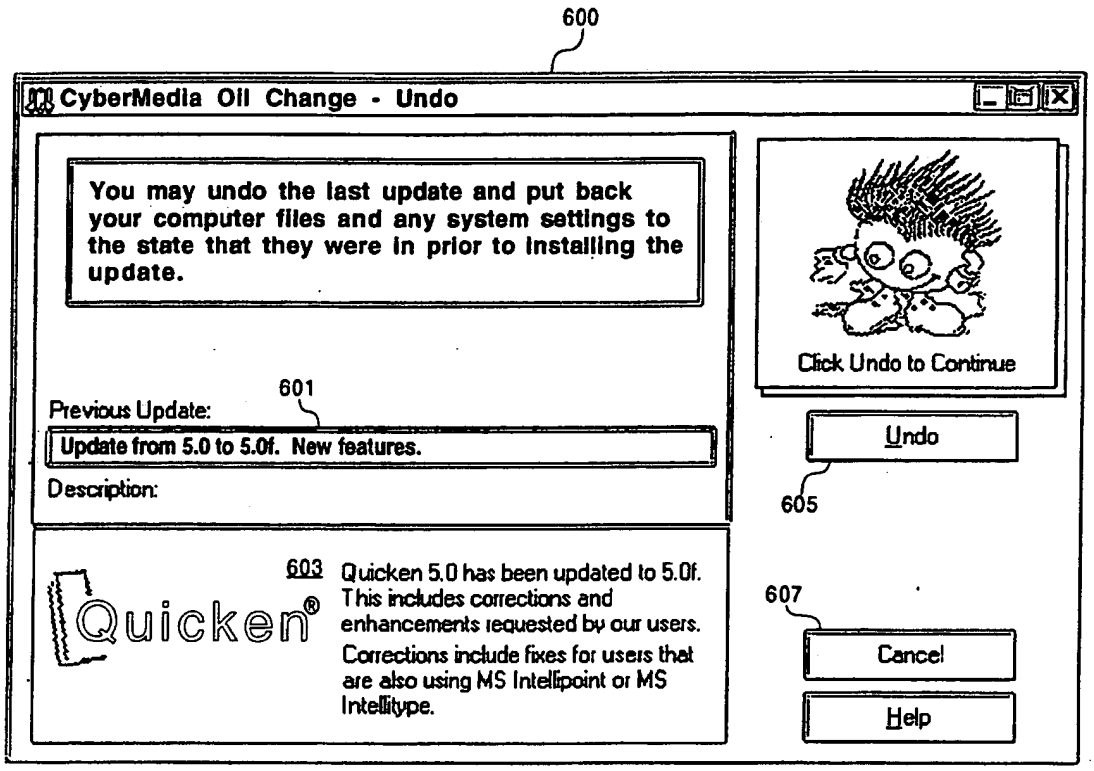

600
CyberMedia Oil Change - Undo
You may undo the last update and put back your computer files and any system settings to the state that they were in prior to installing the update.
Click Undo to Continue
601
Previous Update:
Update from 5.0 to 5.0f. New features.
Description:
Undo
605
603
Quicken®
Quicken 5.0 has been updated to 5.0f. This includes corrections and enhancements requested by our users. Corrections include fixes for users that are also using MS Intellipoint or MS Intellitype.
607
Cancel
Help

FIGURE 6

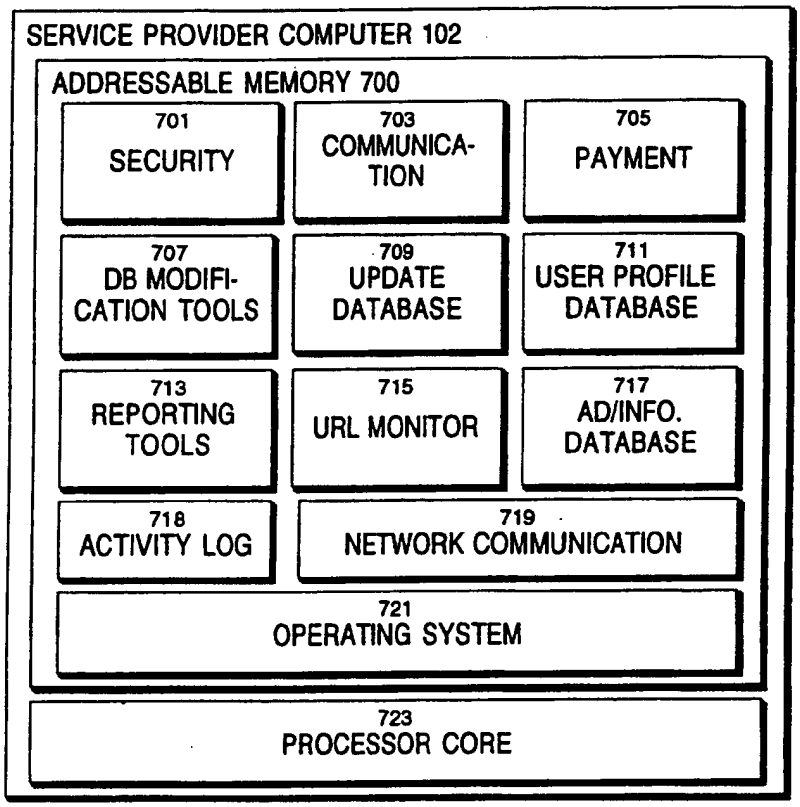

SERVICE PROVIDER COMPUTER 102
ADDRESSABLE MEMORY 700
701
SECURITY
703
COMMUNICA-TION
705
PAYMENT
707
DB MODIFI-CATION TOOLS
709
UPDATE DATABASE
711
USER PROFILE DATABASE
713
REPORTING TOOLS
715
URL MONITOR
717
AD/INFO. DATABASE
718
ACTIVITY LOG
719
NETWORK COMMUNICATION
721
OPERATING SYSTEM
723
PROCESSOR CORE

FIGURE 7

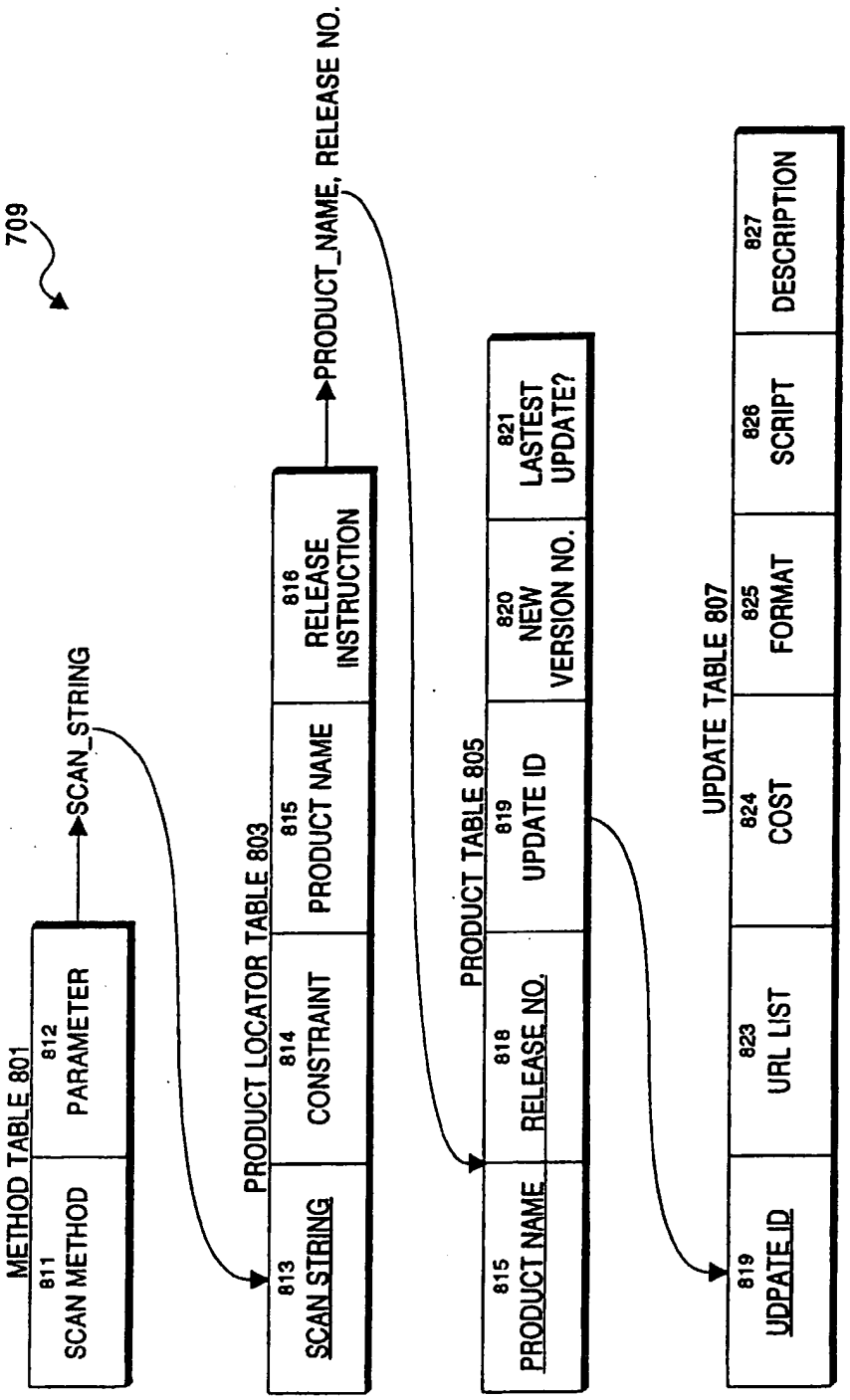

709
METHOD TABLE 801
811 SCAN METHOD
812 PARAMETER
SCAN_STRING
PRODUCT LOCATOR TABLE 803
813 SCAN STRING
814 CONSTRAINT
815 PRODUCT NAME
816 RELEASE INSTRUCTION
PRODUCT_NAME, RELEASE NO.
PRODUCT TABLE 805
815 PRODUCT NAME
818 RELEASE NO.
819 UPDATE ID
820 NEW VERSION NO.
821 LASTEST UPDATE?
UPDATE TABLE 807
819 UPDATE ID
823 URL LIST
824 COST
825 FORMAT
826 SCRIPT
827 DESCRIPTION

FIGURE 8

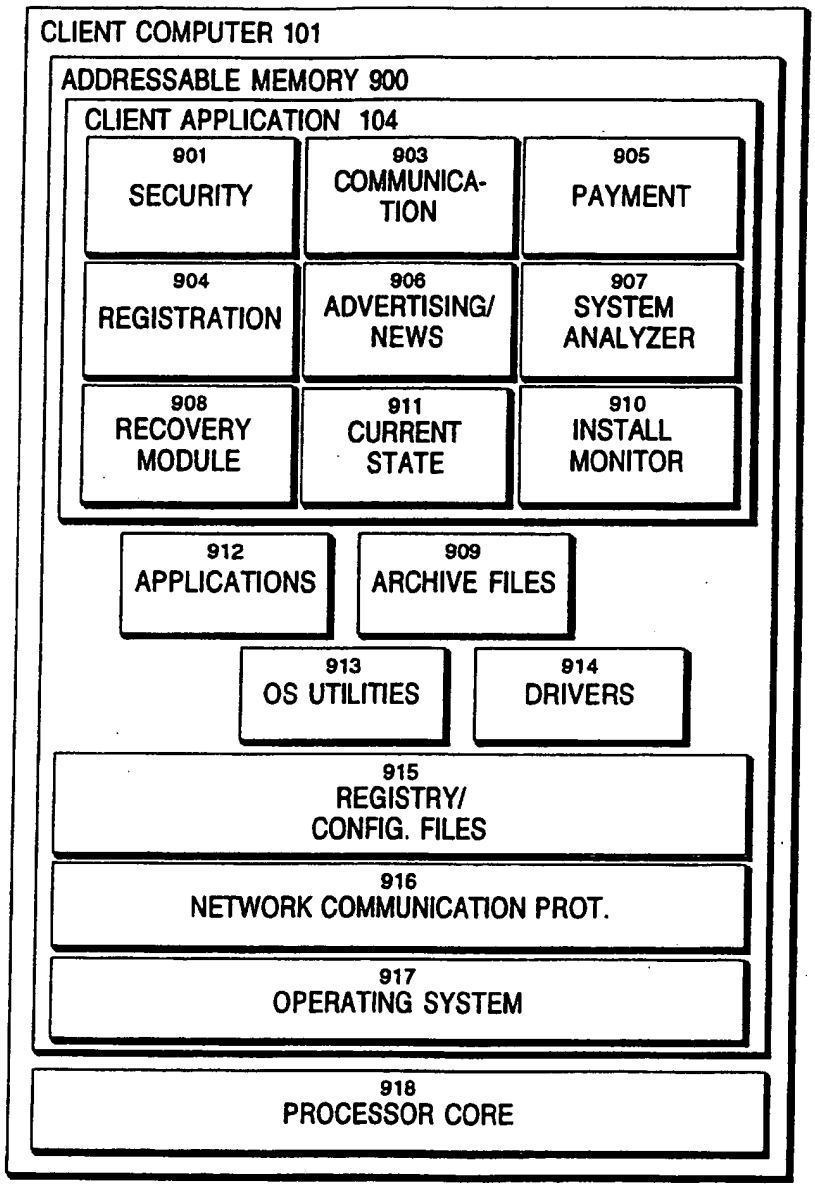

CLIENT COMPUTER 101
ADDRESSABLE MEMORY 900
CLIENT APPLICATION 104
901 SECURITY
903 COMMUNICA-TION
905 PAYMENT
904 REGISTRATION
906 ADVERTISING/ NEWS
907 SYSTEM ANALYZER
908 RECOVERY MODULE
911 CURRENT STATE
910 INSTALL MONITOR
912 APPLICATIONS
909 ARCHIVE FILES
913 OS UTILITIES
914 DRIVERS
915 REGISTRY/ CONFIG. FILES
916 NETWORK COMMUNICATION PROT.
917 OPERATING SYSTEM
918 PROCESSOR CORE

FIGURE 9

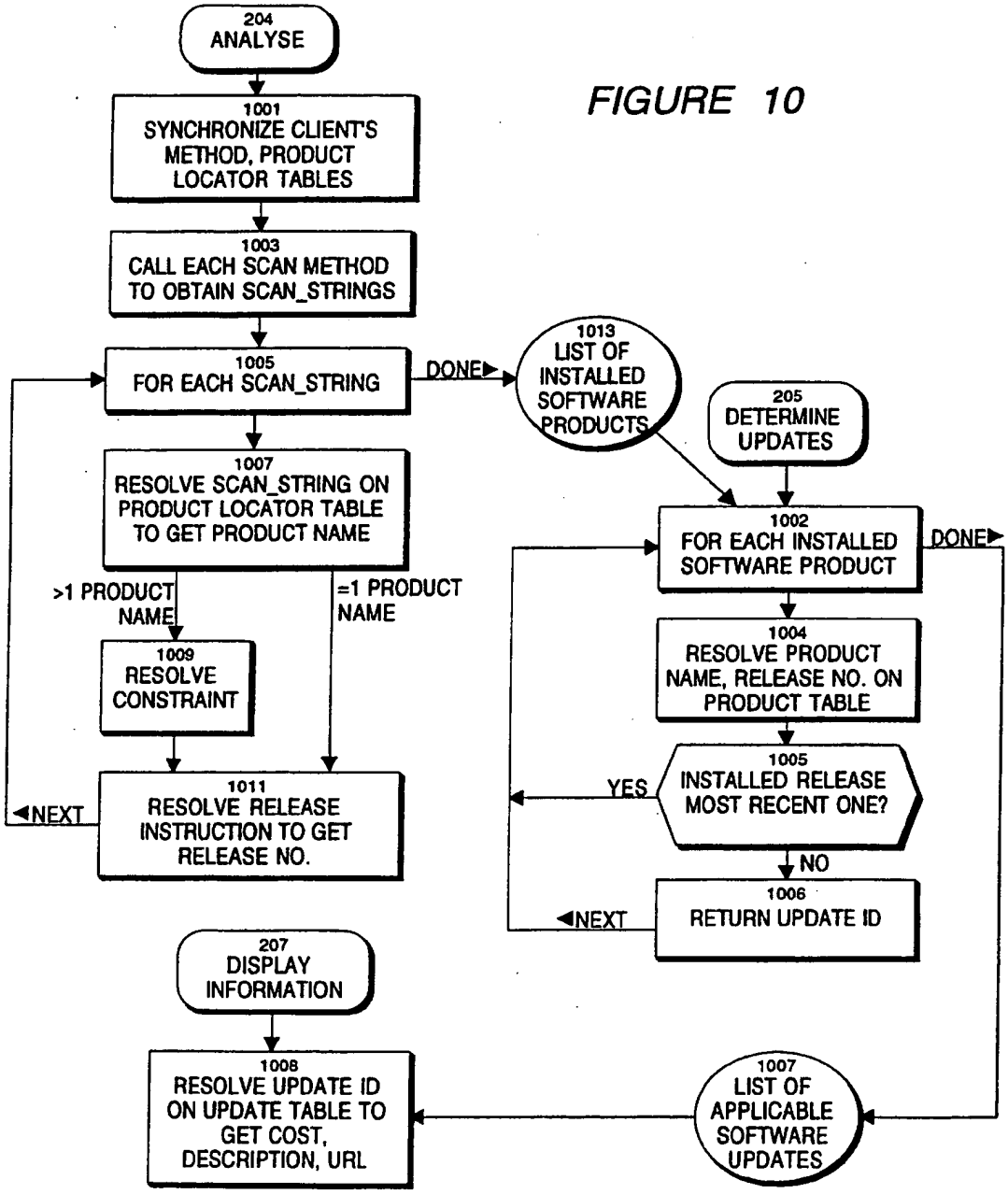
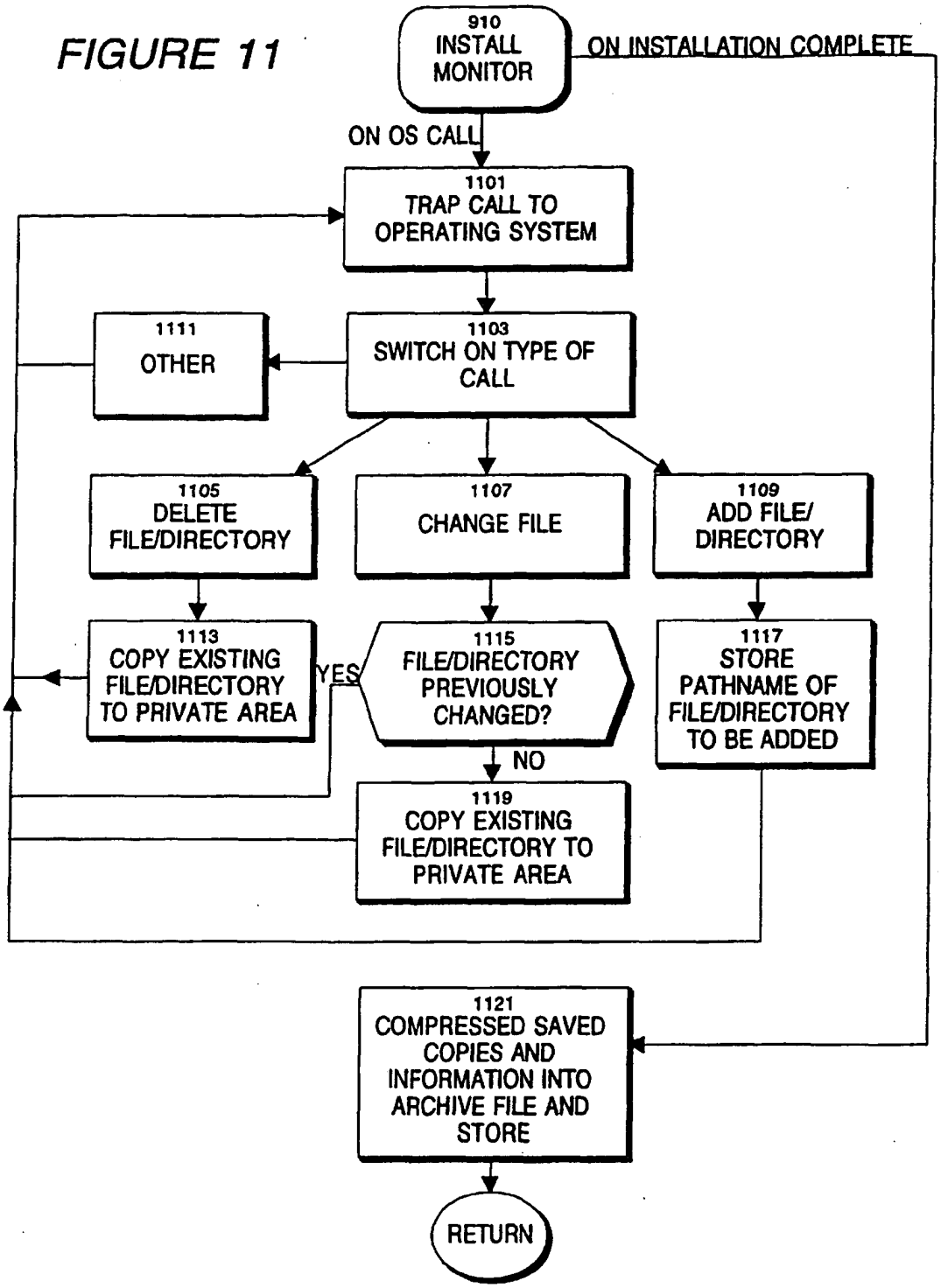

204
ANALYSE
1001
SYNCHRONIZE CLIENT'S METHOD, PRODUCT LOCATOR TABLES
1003
CALL EACH SCAN METHOD TO OBTAIN SCAN_STRINGS
1005
FOR EACH SCAN_STRING
DONE
1007
RESOLVE SCAN_STRING ON PRODUCT LOCATOR TABLE TO GET PRODUCT NAME
>1 PRODUCT NAME
=1 PRODUCT NAME
1009
RESOLVE CONSTRAINT
1011
RESOLVE RELEASE INSTRUCTION TO GET RELEASE NO.
NEXT
1013
LIST OF INSTALLED SOFTWARE PRODUCTS
205
DETERMINE UPDATES
1002
FOR EACH INSTALLED SOFTWARE PRODUCT
DONE
1004
RESOLVE PRODUCT NAME, RELEASE NO. ON PRODUCT TABLE
1005
INSTALLED RELEASE MOST RECENT ONE?
YES
NO
1006
RETURN UPDATE ID
NEXT
207
DISPLAY INFORMATION
1008
RESOLVE UPDATE ID ON UPDATE TABLE TO GET COST, DESCRIPTION, URL
1007
LIST OF APPLICABLE SOFTWARE UPDATES

FIGURE 10

FIGURE 11


910
INSTALL MONITOR
ON INSTALLATION COMPLETE
ON OS CALL
1101
TRAP CALL TO OPERATING SYSTEM
1111
OTHER
1103
SWITCH ON TYPE OF CALL
1105
DELETE FILE/DIRECTORY
1107
CHANGE FILE
1109
ADD FILE/ DIRECTORY
1113
COPY EXISTING FILE/DIRECTORY TO PRIVATE AREA
YES
1115
FILE/DIRECTORY PREVIOUSLY CHANGED?
1117
STORE PATHNAME OF FILE/DIRECTORY TO BE ADDED
NO
1119
COPY EXISTING FILE/DIRECTORY TO PRIVATE AREA
1121
COMPRESSED SAVED COPIES AND INFORMATION INTO ARCHIVE FILE AND STORE
RETURN

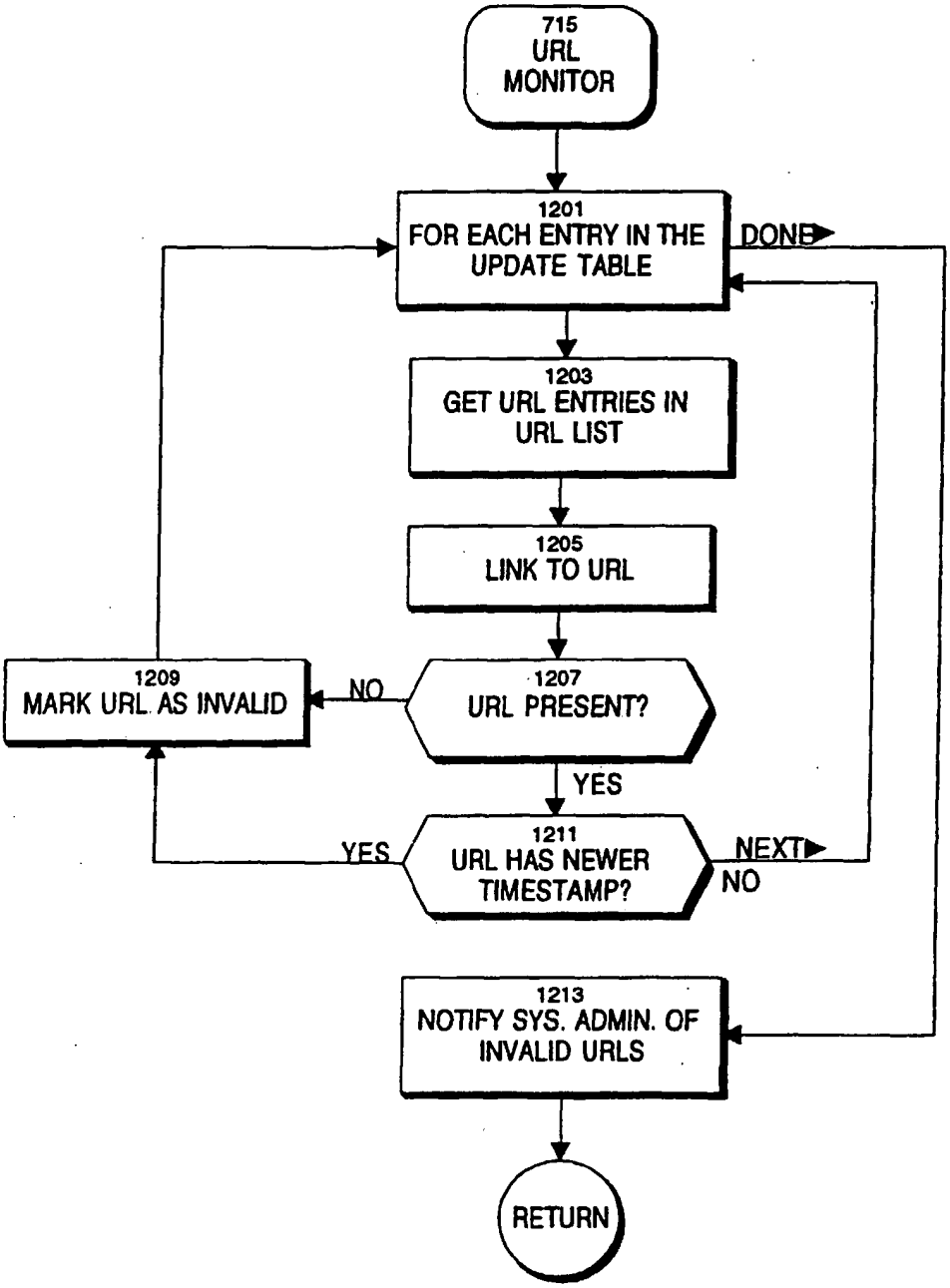
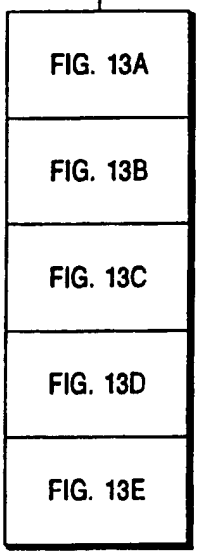

715
URL
MONITOR
1201
FOR EACH ENTRY IN THE
UPDATE TABLE
DONE
1203
GET URL ENTRIES IN
URL LIST
1205
LINK TO URL
1207
URL PRESENT?
NO
1209
MARK URL AS INVALID
YES
1211
URL HAS NEWER
TIMESTAMP?
YES
NEXT
NO
1213
NOTIFY SYS. ADMIN. OF
INVALID URLS
RETURN

FIGURE 12

FIGURE 13





Welcome to the *Oil Change* Software Update Page!

DESCRIBE THE UPDATE

Please provide a brief description of this update:
1301

If available, provide URL where additional information regarding this update would be available
1303

What is the new version number?
1305
What is the new version number?

Which product does this update affect?
(If you are listing multiple products, please separate product names with a comma)
1307

Oil Change allows you to choose one of four different updates:

A Replacement Update affects a specific version already on the PC.
A Cumulative Update does not require an existing version on the PC.
A Point Update installs components without changing the version number.
An Incremental Update only modifies certain portions of a specific version.

Which type of update is this?
Replacement Update
1309

Please list any known incompatibilities
1311

Figure 13a

SPECIFY WHEN THIS UPDATE IS APPLICABLE

In order to verify that the correct product/version is updated, *Oil Change* provides a series of filters for use in establishing additional file criteria. If necessary, you may select one or more of these filters:

☐ **LOCATE PREVIOUS VERSION OF THE FILE**
Check here if you want the program to scan the full system for previous versions

What is the name of the file?
[]

1313

Where is the file located?
[Root Directory (C:\)]

Which version should Oil Change look for?
[Specific Version]

If you selected Specific Version,
enter version to be verified: []

If you selected Multiple Versions,
enter the version numbers: []
(for multiple versions, enter the version numbers separated by a comma, i.e. 1.01, 1.02, etc.)

If you selected Version Range,
enter versions in the range From: [] To: []
(If searching for all subsequent versions, enter INFINITE in the To: field)

☐ **CHECK DATE OF PREVIOUSLY INSTALLED FILE**
Select this option if you would like the program to scan the system for the dates of previous versions

Enter name of file
[]

1315

Where is the file located?
[Root Directory (C:\)]

Which date should Oil Change look for?
[Look for a Specific File Date]

Figure 13b

If you chose a Specific Date,
enter date here (mm/dd/yy)

If you selected Latest Date or Date Range, specify the range
(inclusive) to be checked From: To:
(to search for any date prior to the To: field, specify 00/00/00 in the From: field)

☐ **CHECK REGISTRY**
Select this option if you would like the program to check the Registry for product information

What is the MajorKey in the Registry?
HKEY_CLASSES_ROOT

1317

What is the MinorKey in the Registry?
(i.e. System\CurrentControlSet\Control\MediaResources)

What is the Registry SubKey?
*(Enter * if all SubKeys may be used)*

What is the Registry SubKey Value Name?
(Name of the value in the SubKey, e.g. installer, driver, etc.)

What is the Registry SubKey Value Type?
Specific Version

If you selected Specific Version,
enter the version number

If you selected Version Range,
enter versions in the range From: To:
(If searching for all subsequent versions, enter INFINITE in the To: field)

If you selected Specific Date,
enter date here (mm/dd/yy)

If you selected Date Range,
enter range here From: To:
(to search for any date prior to the To: field, specify 00/00/00 in the From: field)

If you selected Specific String,
how should the strings be compared?
Compare String - CASE SENSITIVE

Figure 13c

Enter the full Registry String

(enter the identifying string your product creates in the Windows Registry)

WHERE CAN THE UPDATE BE DOWNLOADED FROM?

Enter the URL of the site where the file is available for download (If there are multiple files, use a comma to seperate the URLs)

1319

What file format is being used?

Self Extracting File 1321

☐ *Check here if you would like CyberMedia to store your update on our local FTP site.*

Tell us how to install your product

Specify the installation steps in the order they are to be performed. Please note that some actions will require additional input. Once you have made a selection from the list box, follow the instructions noted on the list entry.

INSTALLATION - STEP 1

Select one: Execute INSTALL.EXE 1323

A-SOURCE/FILE NAME

B-SOURCE DIRECTORY NAME: Root Directory (C:\)

C-DEST. DIRECTORY NAME: Root Directory (C:\)

D-COMMAND LINE PARAMETERS (optional)

INSTALLATION - STEP 2

Select one: Execute INSTALL.EXE

A-SOURCE/FILE NAME

B-SOURCE DIRECTORY NAME: Root Directory (C:\)

C-DEST. DIRECTORY NAME: Root Directory (C:\)

D-COMMAND LINE PARAMETERS (optional)

INSTALLATION - STEP 3

Select one: Execute INSTALL.EXE

A-SOURCE/FILE NAME

B-SOURCE DIRECTORY NAME: Root Directory (C:\)

C-DEST. DIRECTORY NAME: Root Directory (C:\)

D-COMMAND LINE PARAMETERS (optional)

Figure 13d

INSTALLATION - STEP 4

Select one: Execute INSTALL.EXE

A-SOURCE/FILE NAME

B-SOURCE DIRECTORY NAME: Root Directory (C:\)

C-DEST. DIRECTORY NAME: Root Directory (C:\)

D-COMMAND LINE PARAMETERS (optional)

Return to Product Registration page

Return to Press Kit

We welcome your comments and suggestions

©Copyright CyberMedia, Inc 1996. All Rights Reserved.

Figure 13e

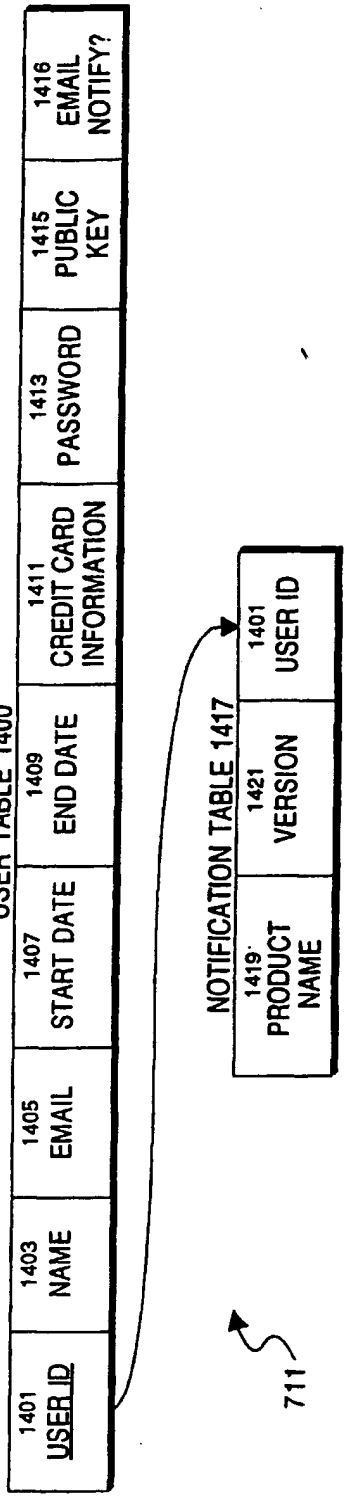


FIGURE 14

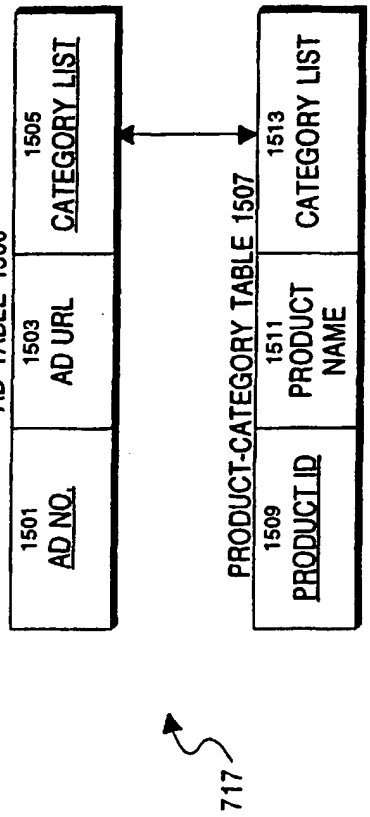


FIGURE 15

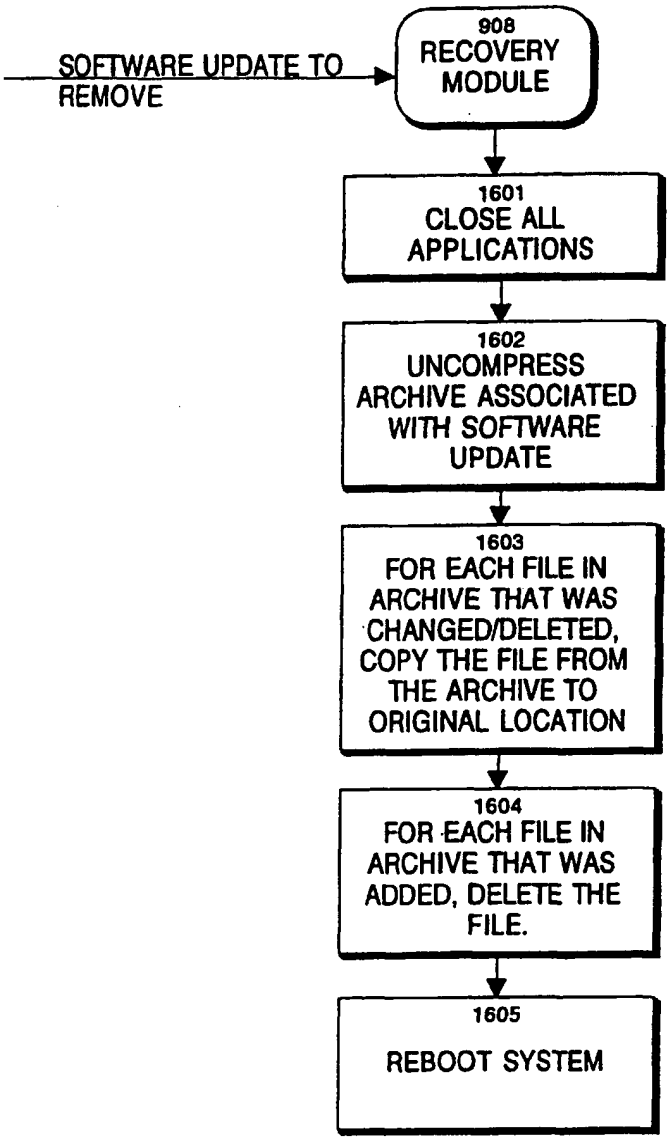
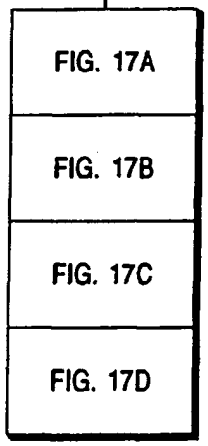

SOFTWARE UPDATE TO REMOVE
908
RECOVERY MODULE
1601
CLOSE ALL APPLICATIONS
1602
UNCOMPRESS ARCHIVE ASSOCIATED WITH SOFTWARE UPDATE
1603
FOR EACH FILE IN ARCHIVE THAT WAS CHANGED/DELETED, COPY THE FILE FROM THE ARCHIVE TO ORIGINAL LOCATION
1604
FOR EACH FILE IN ARCHIVE THAT WAS ADDED, DELETE THE FILE.
1605
REBOOT SYSTEM

FIGURE 16

FIGURE 17



CyberMedia 1700

Welcome to *Oil Change* Product Registration!

This is an example of the product registration form to be used for *Oil Change*. We welcome your comments and suggestions.

PRODUCT INFORMATION

Company Name: 1701

Product Name: 1703

What type of product is this? 1705

Device Driver

How do you uniquely identify this product on a system?

Signature File 1707

Identifying file name or character string: 1709

*If you would like **Oil Change** to use additional product verification filters, please select one or more of the following:*

Check Version

Check File Date

Check Registry Entry

Check for Directory

☐ **CHECK VERSION**

Select this option if you would like the program to scan the system for previous versions

Figure 17a

What is the name of the file?

1709

Where is this file located?

Root Directory (C:\)

Check for which version?

Any Version

If you selected Specific Version,
which version should we verify?

If you selected Multiple Versions,
enter the version numbers:
(for multiple versions, enter the version numbers separated by a comma, i.e. 1.01, 1.02,

If you selected Version Range,
enter versions in the range From: To:
(If searching for all subsequent versions, enter INFINITE in the To: field)

☐ **CHECK FILE DATE**

Select this option if you would like the program to scan the system for the dates of previous versions

What is the name of the file?

1711

Where is this file located?

Root Directory (C:\)

Which date should Oil Change check for?

Look for a Specific File Date

If you chose a Specific Date,
enter date here (mm/dd/yy)

If you selected Latest Date, what range
will the date fall between? From: To:
(to search for any date prior to the To: field, specify 00/00/00 in the From: field)

Figure 17b

If you selected **Version Range**, enter versions
in the range From: [] To: []
(If searching for all subsequent versions, enter INFINITE in the To: field)

If you selected **Specific Date**,
enter date here (mm/dd/yy) []

If you selected **Date Range**,
enter range here From: [] To: []
(to search for any date prior to the To: field, specify 00/00/00 in the From: field)

If you selected **Specific String**,
how should the strings be compared?
[Compare String - CASE SENSITIVE]

Enter the full Registry String
[]
(enter the identifying string your product creates in the Windows Registry)

☐ **CHECK FOR EXISTING DIRECTORY**
Select this option if you would like the program to scan for an existing directory

1713

If you selected **Version Range**, enter versions
in the range From: [] To: []
(If searching for all subsequent versions, enter INFINITE in the To: field)

If you selected **Specific Date**,
enter date here (mm/dd/yy) []

If you selected **Date Range**,
enter range here From: [] To: []
(to search for any date prior to the To: field, specify 00/00/00 in the From: field)

If you selected **Specific String**,
how should the strings be compared?
[Compare String - CASE SENSITIVE]

Enter the full Registry String
[]
(enter the identifying string your product creates in the Windows Registry)

☐ **CHECK FOR EXISTING DIRECTORY**
Select this option if you would like the program to scan for an existing directory

Figure 17c

Which directory name should be verified?

Where is the directory located?

Root Directory (C:\)

(Note: If you have multiple programs using the same information, you must submit a separate verification form for each. After submitting the initial form, use your browser's "back" button to return to this page, then click "Clear Entry" to reset the form. You may then enter new data for additional products.)

View the Update Registration page

Return to Press Kit

We welcome your comments and suggestions

©Copyright CyberMedia, Inc 1996. All Rights Reserved.

Figure 17d

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 822 473 A3

(12) EUROPEAN PATENT APPLICATION

(88) Date of publication A3:
11.11.1998 Bulletin 1998/46

(51) Int Cl.[6]: **G05B 19/418**

(43) Date of publication A2:
04.02.1998 Bulletin 1998/06

(21) Application number: **97305732.6**

(22) Date of filing: **30.07.1997**

(84) Designated Contracting States:
AT BE CH DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE

(30) Priority: **31.07.1996 JP 202057/96**
24.09.1996 JP 251623/96
24.06.1997 JP 167233/97

(71) Applicant: **CANON KABUSHIKI KAISHA**
Tokyo (JP)

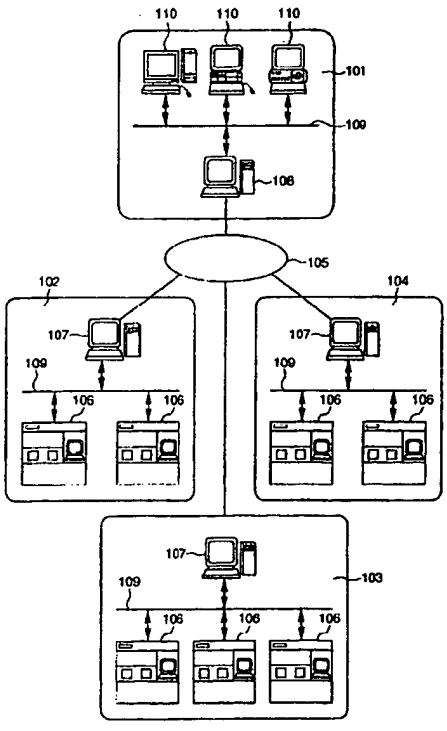
(72) Inventors:
- **Ogushi, Nobuaki**
 Ohta-ku, Tokyo (JP)
- **Ohta, Hirohisa**
 Ohta-ku, Tokyo (JP)
- **Yoneyama, Yoshito**
 Ohta-ku, Tokyo (JP)
- **Ogura, Masaya**
 Ohta-ku, Tokyo (JP)

(74) Representative:
Beresford, Keith Denis Lewis et al
BERESFORD & Co.
2-5 Warwick Court
High Holborn
London WC1R 5DJ (GB)

(54) Remote maintenance system

(57) Factories (102 - 104) have host computers (107) for monitoring industrial equipments (106). Each host computer (107) is connected to a management host computer (108) on a vendor (101) side through the internet (105). The host computer (107) on the factory side detects occurrence of a trouble of the industrial equipment (106) and notifies the vendor side of status information representing a trouble state. In response to this, the host computer (108) on the vendor side notifies the factory side of response information representing a countermeasure against the trouble state.

FIG. 1



EP 0 822 473 A3

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 97 30 5732

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
A	EP 0 051 861 A (KEARNEY & TRECKER CORP) 19 May 1982 * the whole document * ---	1-30	G05B19/418
A	US 5 212 645 A (WILDES DOUGLAS G ET AL) 18 May 1993 * the whole document * ---	1-30	
A	DE 195 06 764 A (SANYO ELECTRIC CO) 31 August 1995 * the whole document * ---	1-30	
A	US 5 315 711 A (BARONE STAN T ET AL) 24 May 1994 * the whole document * ---	1-30	
A	FR 2 688 643 A (BALI DEV) 17 September 1993 * the whole document * -----	1-30	
			TECHNICAL FIELDS SEARCHED (Int.Cl.6) G05B

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	23 September 1998	Hauser, L

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 811 942 A3

(12) EUROPEAN PATENT APPLICATION

(88) Date of publication A3:
10.02.1999 Bulletin 1999/06

(43) Date of publication A2:
10.12.1997 Bulletin 1997/50

(21) Application number: **97109222.6**

(22) Date of filing: **06.06.1997**

(51) Int. Cl.[6]: **G06F 17/60**

(84) Designated Contracting States:
AT BE CH DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE

(30) Priority: **07.06.1996 US 660488**

(71) Applicant:
Cyber Media, Incorporated
Santa Monica, CA 90405 (US)

(72) Inventors:
- **Cheng, William**
 San Gabriel, CA. 91775 (US)
- **Hwang, Kenneth**
 Los Angeles, CA. 90049 (US)
- **Kannan, Ravi**
 Los Angeles, CA. 90024 (US)
- **Katchapalayam, Babu**
 Los Angeles, CA. 90049 (US)
- **Liu, Bing**
 Alhambra, CA. 91801 (US)
- **Narasimhan, Balaji**
 Culver City, CA. 90230 (US)
- **Ramanujam, Gopal**
 Los Angeles, CA 90034 (US)
- **Tran, Jonathan**
 Alhambra, CA. 91803 (US)

(74) Representative:
Liesegang, Roland, Dr.-Ing. et al
FORRESTER & BOEHMERT
Franz-Joseph-Strasse 38
80801 München (DE)

(54) Automatic updating of diverse software products on multiple client computer system

(57) A system and method update client computers of various end users with software updates for software products installed on the client computers, the software products manufacturered by diverse, unrelated software vendors. The system includes a service provider computer system, a number of client computers and software vendor computer systems communicating on a common network. The service provider computer system stores in an update database information about the software updates of the diverse software vendors, identifying the software products for which software updates are available, their location on the network at the various software vendor computer systems, information for identifying in the client computers the software products stored thereon, and information for determining for such products, which have software updates available. Users of the client computers connect to the service provider computer and obtain a current version of portions of the database. The client computer determines that software products stored thereon, and using this information, determines from the database, which products have updates available, based on product name and release information for the installed products. The user selects updates for installation. The selected updates are downloaded from the software vendor computer systems and installed on the client computer. Payment for the software update and the service is mediated by the service provider computer. Authentication of the user ensures only registered users obtain software updates. Authentication of the software updates ensures that the software updates are virus free and uncorrupted. Changes to the client computer during installation are monitored and archived, allowing the updates to be subsequently removed by the user.

EP 0 811 942 A3

Printed by Xerox (UK) Business Services

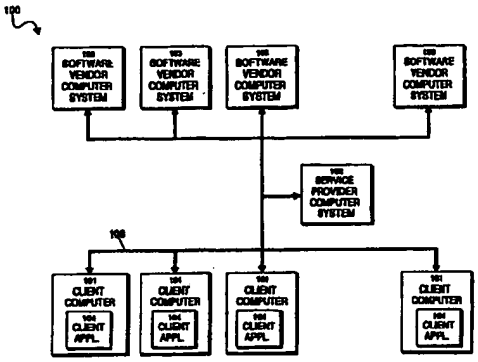

100
SOFTWARE VENDOR COMPUTER SYSTEM
SOFTWARE VENDOR COMPUTER SYSTEM
SOFTWARE VENDOR COMPUTER SYSTEM
SOFTWARE VENDOR COMPUTER SYSTEM
SERVICE PROVIDER COMPUTER SYSTEM
106
CLIENT COMPUTER
CLIENT APPL.
CLIENT COMPUTER
CLIENT APPL.
CLIENT COMPUTER
CLIENT APPL.
CLIENT COMPUTER
CLIENT APPL.

FIGURE 1

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 97 10 9222

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.6)
X	WO 95 34857 A (SMITH JAMES P ;SMITH EDWARD A (US)) 21 December 1995 * page 20, line 4 - page 23, line 14 * * page 8, line 6 - page 11, line 12 * * page 6, line 25 - page 7, line 18 * * abstract; claims 1,5-13,15,20,26; figure 2 *	1-6, 9-21, 23-26	G06F17/60
Y		7,8,22	
Y	KELLER A M: "Smart catalogs and virtual catalogs" PROCEEDINGS OF THE FIRST USENIX WORKSHOP OF ELECTRONIC COMMERCE, PROCEEDINGS OF THE FIRST USENIX WORKSHOP OF ELECTRONIC COMMERCE, NEW YORK, NY, USA, 11-12 JULY 1995, pages 125-131, XP002086428 1995, Berkeley, CA, USA, USENIX Assoc, USA * page 125, column 2, paragraph 2 - page 130, column 1, paragraph 2 * * abstract; figure 1 *	7,8,22	TECHNICAL FIELDS SEARCHED (Int.Cl.6)
A	ANONYMOUS: "Remote Subscription Services" IBM TECHNICAL DISCLOSURE BULLETIN, vol. 37, no. 6B, June 1994, pages 263-264, XP002086429 New York, US * the whole document *	1-26	G06F

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	2 December 1998	Gardiner, A

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 97 10 9222

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

02-12-1998

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 9534857 A	21-12-1995	AU 2774495 A	05-01-1996

EPO FORM P0459